Reference /

(19) JAPANESE PATENT OFFICE

PATENT ABSTRACTS OF JAPAN

(11) Publication number: 09200269 A

(43) Date of publication of application: 31.07.97

(51) Int. Cl H04L 12/56
H04L 12/28

(21) Application number: 08349409

(22) Date of filing: 27.12.96

(30) Priority: 29.12.95 US 95 581683

(71) Applicant: AT & T CORP

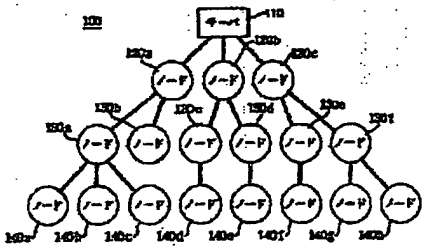
(72) Inventor: OTTO GEORGE V E

(54) SYSTEM AND METHOD FOR PROPAGATING REVISION THROUGH COMMUNICATION NETWORK

(57) Abstract

PROBLEM TO BE SOLVED: To obtain a system for propagating a program or data through a network without reducing or consuming server's resources by providing the system with 1st and 2nd information revision circuits and a status reporting circuit.

SOLUTION: The revision of at least a part of information stored in a server 110 is propagated through the communication network 100 based upon its level. Each of 2nd level nodes 120a to 120c collects and transmits the current status of one or more 2nd level information. A server 110 is a 1st level node, receives the current status of 2nd level information and determines whether revision of one or more 2nd level node information is necessary or not by using the current status of the 2nd level information as a function. When the revision is necessary, the server 110 transmits the 2nd level node information to one or more 2nd level nodes 120a to 120c.



(19)日本国特許庁（ＪＰ）　(12)公開特許公報(A)　(11)特許出願公開番号
特開平9－200269
(43)公開日　平成９年(1997)７月31日

(51)Int.Cl.⁶	識別記号	庁内整理番号	ＦＩ		技術表示箇所
Ｈ０４Ｌ　12/56		9466－5K	Ｈ０４Ｌ　11/20	１０２Ａ	
12/28			11/00	３１０Ｚ	

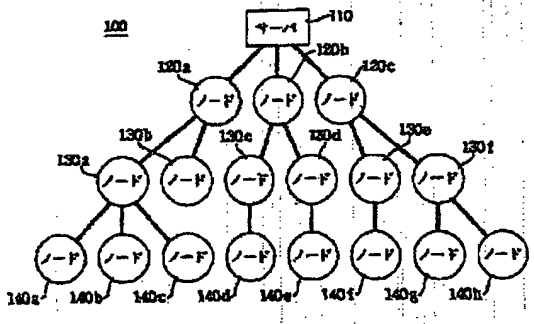
審査請求　未請求　請求項の数21　ＯＬ　（全 36 頁）

(21)出願番号　特願平8－349409
(22)出願日　平成８年(1996)12月27日
(31)優先権主張番号　０８／５８１６８３
(32)優先日　1995年12月29日
(33)優先権主張国　米国（ＵＳ）

(71)出願人　390035493
エイ・ティ・アンド・ティ・コーポレーション
ＡＴ＆Ｔ　ＣＯＲＰ．
アメリカ合衆国　10013－2412　ニューヨーク　ニューヨーク　アヴェニュー　オブ　ジ　アメリカズ　32
(72)発明者　ジョージ　ヴィ．イー．オットー
アメリカ合衆国．07933　ニュージャーシィ．ギレット．ロング　ヒル　ロード　582
(74)代理人　弁理士　岡部　正夫　（外１名）

(54)【発明の名称】　通信ネットワークを通って改訂を伝播するためのシステムと方法

(57)【要約】
【課題】　通信ネットワークのノードに更新を分配するシステムと方法を提供する。
【解決手段】　システムは、（１）第２のノードのメモリに蓄積された第２のノード情報の現在の状態を収集し送信するために、第２の通信ネットワークの第２のノードに関連付けされた状態報告回路、（２）第２のノードから現在の状態を受信し、現在の状態の関数として第２のノード情報の改訂が必要かどうかを決定し、また改訂が必要である場合には、第２のノード情報を改訂するために改訂を第２のノードに送信するために、通信ネットワークの第１のノードに関連付けされた第１の情報改訂回路、（３）通信ネットワークの第３のノードから現在の状態を受信し、第３のノードから前記現在の状態の関数として第３のノードのメモリ内に蓄積された第３のノードの改訂が必要かどうかを決定し、通信ネットワークの前記第２のノードに関連付けされた第２の情報改訂回路を含んでいる。



(19)日本国特許庁（ＪＰ）　(12)公開特許公報(A)　(11)特許出願公開番号

特開平9－200269

(43)公開日　平成9年(1997)7月31日

(51)Int.Cl.⁶	識別記号	庁内整理番号	FI		技術表示箇所
H04L 12/56		9466－5K	H04L 11/20	102A	
12/28			11/00	310Z	

審査請求　未請求　請求項の数21　OL　(全 36 頁)

(21)出願番号　特願平8－349409

(22)出願日　平成8年(1996)12月27日

(31)優先権主張番号　08／581683

(32)優先日　1995年12月29日

(33)優先権主張国　米国（US）

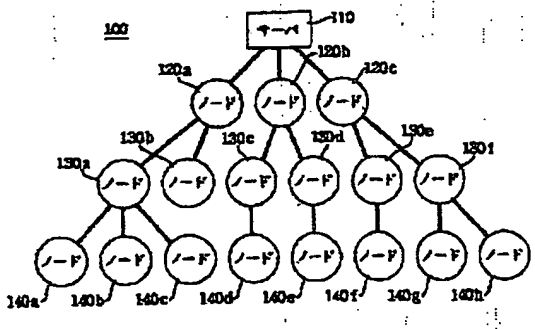
(71)出願人　390035493
エイ・ティ・アンド・ティ・コーポレーション
AT&T CORP.
アメリカ合衆国 10013－2412 ニューヨーク ニューヨーク アヴェニュー オブ アメリカズ 32

(72)発明者　ジョージ ヴィ. イー. オットー
アメリカ合衆国. 07933 ニュージャーシィ, ギレット, ロング ヒル ロード 582

(74)代理人　弁理士　岡部　正夫　(外1名)

(54)【発明の名称】　通信ネットワークを通って改訂を伝播するためのシステムと方法

(57)【要約】
【課題】　通信ネットワークのノードに更新を分配するシステムと方法を提供する。
【解決手段】　システムは、（1）第２のノードのメモリに蓄積された第２のノード情報の現在の状態を収集し送信するために、第２の通信ネットワークの第２のノードに関連付けされた状態報告回路、（2）第２のノードから現在の状態を受信し、現在の状態の関数として第２のノード情報の改訂が必要かどうかを決定し、また改訂が必要である場合には、第２のノード情報を改訂するために改訂を第２のノードに送信するために、通信ネットワークの第１のノードに関連付けされた第１の情報改訂回路、（3）通信ネットワークの第３のノードから現在の状態を受信し、第３のノードから前記現在の状態の関数として第３のノードのメモリ内に蓄積された第３のノードの改訂が必要かどうかを決定し、通信ネットワークの前記第２のノードに関連付けされた第２の情報改訂回路を含んでいる。



1

【特許請求の範囲】

【請求項１】　通信ネットワークを通って改訂を伝播するためのシステムにおいて、

第２のノードのメモリに蓄積された第２のノード情報の現在の状態を収集し送信するために、第２の通信ネットワークの第２のノードに関連付けされた、状態報告回路、

前記第２のノードから前記現在の状態を受信し、前記現在の状態の関数として前記第２のノード情報の改訂が必要かどうかを決定し、また前記改訂が必要である場合には、前記第２のノード情報を改訂するために前記改訂を前記第２のノードに送信するための、前記通信ネットワークの第１のノードに関連付けされた、第１の情報改訂回路、

前記通信ネットワークの第３のノードから現在の状態を受信し、前記第３のノードから前記現在の状態の関数として前記第３のノードのメモリ内に蓄積された第３のノードの改訂が必要かどうかを決定し、前記改訂が必要である場合に、前記第３のノード情報を改訂するために前記第２のノードから受信した改訂を前記第３のノードに送信し、前記改訂が前記第１、第２、および第３ノードを経由し前記通信ネットワークを通って伝播される、前記通信ネットワークの前記第２のノードに関連付けされた、第２の情報改訂回路、

を含むことを特徴とするシステム。

【請求項２】　前記情報改訂回路が加入者リストを蓄積するためのメモリを含み、前記第２の情報改訂回路が前記改訂を前記加入者リストのコンテントの関数として前記改訂を送ることを特徴とする請求項１記載のシステム。

【請求項３】　前記状態報告回路は前記第２のノード情報の前記現在の状態を収集し第１の時において第１のノードに送信し、前記第３のノードに関連付けされた状態情報回路は前記第３のノードから前記現在の状態を収集し第２の時において前記第２のノードに送信し、前記第２の時は、前記第２の情報改訂回路が前記第３のノードに改訂を送信する前に前記第２のノード情報が完全に改訂されるような十分な時間間隔の後に前記第１の時に続くものであることを特徴とする請求項１記載のシステム。

【請求項４】　前記第２の情報改訂回路は、前記第２のノードに関連付けされた第２のプロセッサ上で動作可能な命令のシーケンスから構成され、前記改訂は前記命令のシーケンスの改訂を含むことが可能であり、これにより、前記第２の情報改訂回路の動作を修正できることを特徴とする請求項１記載のシステム。

【請求項５】　前記通信ネットワークは階層的であり、前記第１のノードは前記第２のノードに対してサーバとして機能し、前記第２のノードは前記第３のノードに対してサーバとして機能することを特徴とする請求項１記載のシステム。

2

【請求項６】　前記第１の情報改訂回路は、前記第２のノードから受信した前記現在の状態を、前記第１のノードが前記改訂を前記第２のノードに送信する前に認証するためのセキュリティ回路を含んでおり、前記第２のノードは前記第１のノードから受信した前記改訂を前記第２のノード情報を改訂する前に認証するための第２のセキュリティ回路を含んでいることを特徴とする請求項１記載のシステム。

【請求項７】　前記第１の情報改訂回路は、前記第２のノード上でのロギングにより前記第２のノード情報を改訂し、また前記第２のノードが改訂を受信することを可能とするコマンドのシーケンスを前記第２のノードに送信することを特徴とする請求項１記載のシステム。

【請求項８】　通信ネットワークを通って改訂を伝播するための通信ネットワークの運用方法において、

前記通信ネットワークの第２のノードのメモリ内に蓄積された第２のノード情報の現在の状態を収集し送信するステップ、

前記通信ネットワーク内の第１のノードに前記第２のノードから前記現在の状態を受信するステップであって、前記第１のノードは、前記現在の状態の関数として前記第２のノード情報の改訂が必要かどうかを決定し、また前記改訂が必要である場合には、前記第２のノード情報を改訂するために前記改訂を前記第２のノードに送信するステップ、

前記通信ネットワークの第３のノードから現在の状態を前記第２のノード内に受信するステップであって、前記第２のノードは、前記第３のノードから前記現在の状態の関数として前記第３のノードのメモリ内に蓄積された第３のノードの改訂が必要かどうかを決定し、前記改訂が必要である場合に、前記第３のノード情報を改訂するために前記第２のノードから受信した改訂を前記第３のノードに送信し、前記改訂が前記第１、第２、および第３ノードを経由し前記通信ネットワークを通って伝播されるステップ、

を含むことを特徴とする方法。

【請求項９】　前記第３のノードからの前記現在の情報を受信するステップが、前記第２のノードの前記メモリ内に蓄積された加入者リストのコンテントの関数として前記改訂を送信するステップを含むことを特徴とする請求項８記載の方法。

【請求項１０】　前記第２のノード情報の前記現在の状態が収集され第１の時において第１のノードに送信され、前記現在の情報を前記第３のノードから収集し前記第２のノードに第２の時において送信するステップをさらに含み、前記第２の時は、前記第２の情報改訂回路が前記第３のノードに改訂を送信する前に前記第２のノード情報が完全に改訂されるような十分な時間間隔の後に前記第１の時に続くものであることを特徴とする請求項

8記載の方法。

【請求項11】　前記第2のノードは、前記第2のノードと関連付けされた第2のプロセッサ上で動作可能な命令のシーケンスを含み、前記改訂は前記命令のシーケンスの改訂を含むことが可能であり、これにより、前記第2のノードの動作が変更されることを特徴とする請求項8記載の方法。

【請求項12】　前記通信ネットワークは階層的であり、前記第1のノードは前記第2のノードに対してサーバとして機能し、前記第2のノードは前記第3のノードに対してサーバとして機能することを特徴とする請求項8記載の方法。

【請求項13】　前記第2のノードから受信した前記現在の状態を、前記第1のノードが前記改訂を前記第2のノードに送信する前に認証するステップ、および前記第1のノードから受信した前記改訂を前記第2のノード情報を改訂する前に認証するステップをさらに含むことを特徴とする請求項8記載の方法。

【請求項14】　前記現在の状態を前記第2のノードから前記第1のノードに受信するステップが、前記第2のノード上でのロギングにより前記第2のノード情報を改訂し、また前記第2のノードが改訂を受信することを可能とするコマンドのシーケンスを前記第2のノードに送信するステップを含むことを特徴とする請求項8記載の方法。

【請求項15】　ホスト、第1のレベルのノードおよび第2のレベルのノードを有する階層的な通信ネットワークを通って改訂を伝播するためのシステムにおいて、

第1のレベルのノードのメモリに蓄積された第1のレベルのノード情報の現在の状態を収集し第1の時において送信するために、前記第1のレベルのノードに関連付けされた状態報告回路、

前記第1のレベルのノードから前記現在の状態を受信し、前記第1のレベルのノード情報の改訂が必要かどうかを決定し、また前記改訂が必要である場合には、前記第1のレベルのノード情報を改訂するために前記改訂を前記第1のレベルのノードに送信するために、前記ホストに関連付けされた、第1の情報改訂回路、および前記第2のレベルのノードから第2の時に前記現在の状態を受信し、前記第2のレベルのノードからの前記現在の状態の関数として、前記第2のレベルのノードのメモリ内に記憶された第2のレベルのノード情報の改訂が必要かどうかを決定し、前記改訂が必要である場合には、前記第2のレベルのノード情報を改訂するために前記ホストから受信した前記改訂を前記第2のレベルのノードに送信し、前記第2の時は、前記第2の情報改訂回路が前記第2のレベルのノードに前記改訂を送信する前に前記第1のレベルのノード情報が完全に改訂されるような十分な時間間隔の後に前記第1の時に続くものである第2の情報改訂回路、

を含むことを特徴とするシステム。

【請求項16】　前記第2の情報改訂回路が加入者リストを蓄積するためのリストを含み、前記第2の情報改訂回路が前記改訂を前記加入者リストのコンテントの関数として送信することを特徴とする請求項15記載のシステム。

【請求項17】　前記第2の情報改訂回路は、前記第1のレベルのノードに関連付けされた第2のプロセッサ上で動作可能な命令のシーケンスから構成され、前記改訂は前記命令のシーケンスの改訂を含むことが可能であり、これにより、前記第2の情報改訂回路の動作を修正できることを特徴とする請求項15記載のシステム。

【請求項18】　前記第1の情報改訂回路は、前記第1のレベルのノードから受信した前記現在の状態を、前記ホストが前記改訂を前記第1のレベルのノードに送信する前に認証するためのセキュリティ回路を含んでおり、前記第1のレベルのノードは前記ホストから受信した前記改訂を前記第1のレベルのノード情報を改訂する前に認証するための第2のセキュリティ回路を含んでいることを特徴とする請求項15記載のシステム。

【請求項19】　前記第2のセキュリティ回路は前記改訂をファイル毎に認証することを特徴とする請求項15記載のシステム。

【請求項20】　前記第1の情報改訂回路は、前記第1のレベルのノード上でのロギングにより前記第1のレベルのノード情報を改訂し、また前記第1のノードが改訂を受信することを可能とするコマンドのシーケンスを前記第1のレベルのノードに送信することを特徴とする請求項15記載のシステム。

【請求項21】　通信ネットワークを通って改訂を伝播するためのシステムにおいて、前記通信ネットワークは少なくとも1つの第1のレベルのノード、少なくとも1つの第2のレベルのノードおよび少なくとも1つの第3のレベルのノードを含み、

前記少なくとも1つの第2のレベルのノードのメモリ内に蓄積された第2のレベルの現在の状態の情報を収集し送信するように動作するために、少なくとも1つの第2のレベルのノードに関連付けされた、状態報告回路、

（1）前記少なくとも1つの第2のレベルのノードからの前記第2のレベルの現在の状態の情報を受信し、

（2）前記第2のレベルの現在の状態の情報の関数として、前記少なくとも1つの第2のレベルのノードの情報が必要かどうかを決定し、（3）前記決定に応答して、前記少なくとも1つの第2のレベルのノードの情報を改訂するべく前記少なくとも1つの第2のレベルのノードの情報を前記少なくとも1つの第2のレベルのノードに選択的に送信するための、前記少なくとも1つの第1のレベルのノードに関連付けされた、第1の情報改訂回路、および（1）前記少なくとも1つの第3のレベルのノードから第3のレベルの現在の状態の情報を受信し、

（2）前記第３のレベルの現在の状態の関数として前記少なくとも１つの第３のレベルのメモリ内に蓄積された前記少なくとも１つの第３のレベルのノードの情報の改訂が必要かどうかを決定し、（3）前記決定に応答して、前記少なくとも１つの第３のレベルのノードの情報を改訂するために前記少なくとも１つの第１のレベルのノードから受信した前記改訂を前記少なくとも１つの第３のレベルのノードに選択的に送信し、前記改訂が前記少なくとも１つの第１のレベルのノード、少なくとも１つの第２のレベルのノード、少なくとも１つの第３のレベルのノードを経由し前記通信ネットワークを通って伝播される、前記少なくとも１つの第２のレベルのノードに関連付けされた、第２の情報改訂回路、
を含むことを特徴とするシステム。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、一般的には、通信ネットワークに関し、特に、更新をその階層の関数としてネットワークを通してカスケードする階層通信ネットワークのノードに更新を分配するためのシステムと方法に関する。

【０００２】

【発明が解決しようとする課題】パーソナルコンピュータ（ＰＣｓ）が提供する技術の著しい進歩により、過去においてメインフレームあるいはミニコンピュータによってのみ達成できた役割をＰＣｓが負えるようになってきた。この点に注目して、多くの会社や個々のユーザは市場で利用可能なＰＣｓをその情報処理の必要性に合致するものとして使用することが多くなっている。よって、ＰＣｓが信頼性高く処理を行うことが重要である。コンピュータシステムの障害の許容範囲は、それらのコンピュータシステムへの信頼性のレベルに関することであり、個々のユーザや会社にとっては興味ある問題である。

【０００３】当初は、ＰＣｓは、それぞれ独立したハードウェア、オペレーティングシステム、応用ソフトウェアおよびユーザデータを含んでいる、スタンドアロンの装置であった。ビジネス組織へのＰＣｓの使用の広がりにつれて、しかしながら、データおよびハードウェア資源の共用の必要性が高まり、ローカルエリアネットワーク（「ＬＡＮ」）が生み出されてきた。ＬＡＮ（あるいはそのより地理的に分布された相対物である広域ネットワーク（「ＷＡＮ」））は、（一般的には高速シリアル通信リンクにより）互いにリンクされた多数のＰＣｓ（「クライアント」）を含み、クライアントにプログラムとデータを配信する比較的に高性能のＰＣあるいはマイクロコンピュータ（「サーバ」）の回りで集中化され、また二次記憶装置ユニットやプリンタのようなシステム全域の資源を管理している。

【０００４】ネットワークとして概念は非常に有用であることが証明されているが、２、３の欠点がある。第１に、ネットワークの管理はサーバ内に焦点されているので、ネットワークの全体の性能は、サーバがボトルネックを処理するようになる際に低下してしまう。第２に、プログラムとデータがサーバによりその種々のクライアントに配信されるので、ソフトウェアのプロバイダあるいは売り手がそのプログラムあるいはデータを変更したときはいつも、分布問題が生じる。変更されたプログラムやデータはサーバからクライアントに時間内に、１日のビジネス日以内に分布されなければならない。従来技術の解決法では、サーバ、あるいはサーバにより認識された「ホストコンピュータ」は、変更されたプログラムあるいはデータの「古い」バージョンをサポートしているクライアントのコンピュータのそれぞれを順次トラバースし、また次いで、「新しい」バージョンを実施するために必要なものとしてこれらクライアントのコンピュータを更新する。他の従来技術の解決では、サーバ、あるいはホストコンピュータは各クライアントをトラバースし、サーバのファイルの特定の１つをそれぞれ含むように更新する。

【０００５】従来技術の解決法の特有の問題は、実質的なサーバ、つまりホストが、全てではないが、クライアントコンピュータの多くと通信リンクを確立し、次いでこれらを更新することで処理資源を使い果たすことである。さらに、サーバは更新を行う責務があり、またサーバによりサービスされるクライアントコンピュータの数が増えることで、ネットワークの全体の性能が著しく低下し、サーバが処理のボトルネックとなってしまう。よって、通信ネットワーク、および特にサーバの資源が減じられたり消費されることなしに、通信ネットワークを通ってプログラムあるいはデータを伝播するためのシステムと方法が必要とされる。これを達成できない従来技術の解決法の場合、通信ネットワークのクライアントコンピュータの１つに分配されたソフトウェア製品を更新する際に大きな障害となる。

【０００６】

【課題を解決するための手段】上記した従来技術の欠点に対処するため、本発明は、通信ネットワークを通って改訂（revision）を伝播するための、システム、および動作方法を提供する。通信ネットワークは複数の関連付けされたノードを含んでいる。

【０００７】本発明のシステムは、（1）第２のノードのメモリに蓄積された第２のノード情報の現在の状態を収集し送信するために、第２の通信ネットワークの第２のノードに関連付けされた、状態報告（status report）回路、（2）前記第２のノードから前記現在の状態を受信し、前記現在の状態の関数として前記第２のノード情報の改訂が必要かどうかを決定し、また前記改訂が必要である場合には、前記第２のノード情報を改訂するために前記改訂を前記第２のノードに送信するために、

前記通信ネットワークの第１のノードに関連付けされた、第１の情報改訂回路、（３）前記通信ネットワークの第３のノードから現在の状態を受信し、前記第３のノードから前記現在の状態の関数として前記第３のノードのメモリ内に蓄積された第３のノードの改訂が必要かどうかを決定し、前記改訂が必要である場合に、前記第３のノード情報を改訂するために前記第２のノードから受信した改訂を前記第３のノードに送信し、前記改訂が前記第１、第２、および第３ノードを経由し前記通信ネットワークを通って伝播される、前記通信ネットワークの前記第２のノードに関連付けされた、第２の情報改訂回路を含んで構成される。

【０００８】本発明は、それ故、通信ネットワークを通って自動的に伝播するために改訂を許容する。ネットワーク内のノードは、他のノード内の情報への改訂が必要になったときの検出、および他のノードへの改訂を送信することの両方について責務がある。本明細書において使用される「情報」の用語は、指示（つまり、プログラム、機能、タスク、サブルーチン、処理など）およびデータの両方を包含する広く規定されるものである。本発明により改訂を受ける「情報」は、例えば、コンピュータプログラム（プログラム更新、固定（fix）、ツールなどの自動的な分布を許容する）、コンピュータデータ（例えば、ドキュメント、スプレッドシート、データベース、データファイルなど）、ビデオデータなどである。

【０００９】本発明の１つの実施の形態において、少なくとも第２の情報改訂回路は、加入者リストを蓄積するためのメモリを含んでいる。そして、第２の情報改訂回路は上記の改訂を加入者リストのコンテントの関数（function）として送信する。本発明はそれ故、料金に基づく更新サービスのコアを形成することができ、加入者は更新に対して料金を支払う。更新される情報の量および更新の頻度は選択可能であり、料金に基づくサービスの範囲が提供される。関連する実施の形態においては、加入者リストおよび現在の状態が特定のユーザあるいはユーザグループに利用可能な加入者リストの情報のサブセットを識別するために適切に処理され、処理はそれ故に加入者リストに対するフィルタとして機能する。

【００１０】本発明の１つの実施の形態において、状態報告回路は第２のノード情報の現在の状態を収集し第１の時において第１のノードに送信し、第３のノードに関連付けされた状態情報回路は第３のノードからの現在の状態を収集し第２の時において第２のノードに送信し、第２の時は、第２の情報改訂回路が第３のノードに改訂を送信する前に第２のノード情報が完全に改訂されるような十分な時間間隔の後に第１の時に続くものである。これにより、ネットワークを通っての改訂の「波」が順序付けされる。あるいは、改訂は、１つのノードが他のものの改訂の必要を決定するように、よりランダムな態様で分布される。

【００１１】本発明の１つの実施の形態において、第２の情報改訂回路は、第２のノードに関連付けされた第２のプロセッサ上で動作可能な命令のシーケンスから構成され、改訂は命令のシーケンスの改訂を含むことが可能であり、これにより、第２の情報改訂回路の動作を修正することができる。情報改訂回路それ自体で変更あるいは更新される。

【００１２】本発明の１つの実施の形態において、通信ネットワークは階層的であり、第１のノードは第２のノードに対してサーバとして機能し、第２のノードは第３のノードに対してサーバとして機能する。本発明において、「階層的」の用語は、特定のレベルが他のレベルに対する制御あるいは優先順位を有する（例えば、高次の優先順次レベルが低次の優先順位レベル上にある）、多くのレベルの構造を意味し、第１のレベルのノードは１つまたはそれより多くの第２のレベルのノードの階層的に関連しており、各第２のレベルのノードは１つまたはそれより多くの第３のレベルのノードに階層的に関連しており、各第３のレベルのノードは１つまたはそれより多くの第４のノードに階層的に関連している。優先順位は、配列（例えばシーケンス的に）、責務、機能性などに適宜基づいている。よって、本発明は、広い意味においては、ツリーに基づくネットワーク、およびフラットな、ピアツウピアのネットワークを包含している。本発明はさらに、ＬＡＮやＷＡＮのようなコンピュータネットワークに限定されず、セルラー電話あるいはメッセージページングネットワークのような、無線環境においてシステムのソフトウェアあるいはデータを更新するために電気通信システム内で動作する。

【００１３】本発明の１つの実施の形態において、第１の情報改訂回路は、第２のノードから受信した現在の状態を、第１のノードがこの改訂を第１のノードに送信する前に認証するためのセキュリティ回路を含んでおり、第２のノードは第１のノードから受信した改訂を第２のノード情報を改訂する前に認証するための第２のセキュリティ回路を含んでいる。関連した実施の形態においては、第２のセキュリティ回路はファイル毎（file-by-file）に改訂を認証する。当然のことであるが、セキュリティ回路はコンピュータの命令の形態であり、回路がいつでも変更あるいは更新を行うことができる。

【００１４】本発明の１つの実施の形態において、第１の情報改訂回路は、第２のノード上でのロギングにより第２のノード情報を改訂し、また第２のノードが改訂を受信することを可能とするコマンドのシーケンスを第２のノードに送信する。本発明はさらに、従来のネットワーク環境内で動作し、またそれ故、下層のネットワークオペレーティングシステム（ＮＯＳ）に完全に透過的である。よって、ＮＯＳのセキュリティおよび他の特徴はそのままである。

【００１５】本発明を使用および／または分配するための好ましい実施の形態はソフトウェアとしてである。ソフトウェアの実施の形態においては、適当な従来のメモリあるいは他の同等な記憶媒体に記憶された、複数の命令を含んでいる。これらの命令は、処理回路を有する１つまたはそれより多くのネットワークのノードにより読み出し可能で実行可能である。これらの命令は、実行されたときには、処理回路に通信ネットワークを通って改訂を伝播するように指示する。通信ネットワークは、本発明においては、複数の関連付けされたノードを含んでいる。メモリおよび記憶媒体の例としては、制限されることなしに、磁気記録媒体、光学的記録媒体および半導体記録媒体、あるいはこれらの適宜な組合わせである。

【００１６】上記の点は、本発明の好ましい、択一的な、やや広範囲な特徴であり、当業者には以下の詳細な説明からより良く理解できるものである。本発明の以下に記載した他の特徴は本発明の請求の範囲の主題を構成している。当業者には、開示された概念および特定の実施の形態を本発明の同様な目的を実行するための他の構造の変更に利用することは容易である。当業者にはこのような等価な構造を本発明の技術と範囲を逸脱することなく実現できるものである。

【００１７】

【発明の実施の形態】以下に、本発明の実施の形態を添付図面を参照して説明する。これらの図面において、同じ符号は同様な部品を示している。

【００１８】図１は、本発明の原理が実施される、従来の階層的な通信ネットワークのブロックダイヤグラムを示している。図１を参照して、例示的なネットワーク１００は、サーバノード１１０および複数の従来型のクライアントノード１２０ａ－１２０ｃ、１３０ａ－１３０ｆおよび１４０ａ－１４０ｈを含んでいる。本明細書において、「含む」とは、制限なしのに含むものと定義される。「ノード」の用語は、本明細書において、ポータル（portal）であるなしに拘りなく、通信ネットワーク１００内において、信号、あるいは情報を通信できる能力のある、あらゆる接続部、末端部ないし接続点、ステーション、ターミナルなどを含むものとして定義される。

【００１９】サーバノード１１０は便宜的に、クライアントノード１２０ａ－１２０ｃ、１３０ａ－１３０ｆおよび１４０ｆ－１４０ｈに分割可能である。サーバノード１１０およびクライアントノード１２０ａ－１２０ｃ、１３０ａ－１３０ｆおよび１４０ａ－１４０ｈは好ましくは、通信リンク、ポート装置（例えば、ルータ、ブリッジ、ゲートウェイ、交換機など）などを含む、何らかの従来手段、直接的または間接的に互いに関連している。「関連する」の用語は、本明細書では、内側、相互接続、含む、内側に含む、接続する、結合する、通信可能である、並置する、共動する、介在するなどを含むことを意味する。「または」の用語は、本明細書においては、包含するを意味し、および／またはを意味する。

【００２０】例示した関連するノード１１０、１２０ａ－１２０ｃ、１３０ａ－１３０ｆ、および１４０ａ－１４０ｈは、当業分野では公知のように、各ノードの１つの間において好ましくは資源共有および資源要求の均衡化がされている。各ノードの種々の１つの間の通信は信号の送信と受信を含んでいる。各通信信号はパケット、フレーム、メッセージ、データシーケンスあるいは他の情報運搬の物理量に分割される。信号は典型的には、ノードの種々の１つの間で情報を通信するために使用される不連続なデータ、アドレスあるいは命令オブジェクトのような、関連するデータ項目の集合を含んでいる。

【００２１】以下に図４と図５を参照してより詳しく説明する、好ましい実施の形態においては、サーバ１１０上に蓄積された情報の少なくとも一部の改訂は、本発明の原理にしたがって、通信ネットワーク１００を通って、レベルに基づいて（level-by-level basis）好ましくは伝播される。「改訂」の用語は、本明細書において、変化、変更、付加、削除、調節、交替、変動、カスタム化などを意味するものとして使用される。より詳しくは、第２のレベルのノード１２０ａ－１２０ｃは１つまたはそれより多くの第２のレベルのノード内に蓄積された第２のレベルの情報の現在の状態を収集し送信する。第２のレベルの情報の現在の状態は、全体のレベルあるいは第２のレベルのノード１２０ａ－１２０ｃの個々の１つに対するものである。サーバ１１０は、第１のレベルのノードであり、第２のレベルの情報の現在の状態を受信し、また、第２のレベルの情報の現在の状態を関数として、１つまたはそれより多くの第２のレベルのノードの情報の改訂が必要かどうかを決定する。改訂が必要な場合、サーバ１１０は第２のレベルのノードの情報を１つまたはそれより多くの第２のレベルのノード１２０ａ－１２０ｃに送信する。

【００２２】改訂の後、１つまたはそれより多くの第２のレベルのノード１２０ａ－１２０ｃは少なくとも１つの第３のレベルノード１３０ａ－１３０ｆから第３のレベルの情報の現在のレベルを受信し、また第３のレベルの情報の現在の状態を関数として、１つまたはそれより多くの第３のレベルのノードの情報が必要かどうかを決定する。第３のレベルの情報の現在の状態は同様に全体のレベルまたは第３のレベルのノード１３０ａ－１３０ｆの個々の１つに対するものである。第３のレベルの改訂が必要な場合、１つまたはそれより多くの第２のレベルのノード１２０ａ－１２０ｃはサーバ１１０から受信した改訂の少なくとも一部を１つまたはそれより多くの第３のレベルのノード１３０ａ－１３０ｆに送信し、これにより、第３のレベルのノードの情報が改訂される。

【００２３】上記で特定し説明した実施の形態の重要な特徴は、幅優先タイプないし「ファンアウト」の更新で

ある。より詳しくは、第1のレベルのノードの情報の改訂は第2のレベルのノードの1つ、次いで第3のレベルのノードの1つなどを通って伝播される。そして、第1のレベルのノードの情報に対する改訂は通信ネットワーク100を通って指数的に伝播される。

【0024】例示の処理システムであるPC（全体を符号200で示した）の等大の図を示した図2に戻る。処理システム200は、通信ネットワーク100内の、ノード110、120a－120c、130a－130fおよび140a－140hのいずれかのノードの機能を果たすものである。処理システム200はシャーシ205、ディスプレイ装置210およびキーボード215を含んでいる。シャーシ205は、ハードディスクドライブ220およびフロッピーディスクドライブ225を含んでいる。フロッピーディスク225は、テープドライブおよびコンパクトディスクドライブ、電話システムおよび装置（電話、ビデオ電話、ファクシミリなどを含む）、ネットワーク通信ポートなどを含む、データあるいは命令を転送するための他の従来構造に置き換えあるいは組み合わせることもできる。

【0025】シャーシ205はバッテリー230、クロック235、分離されたローカルメモリ240および処理回路245（CPU）を例示するために部分的に切断されており、これらは全て内部に収容される。蓄積された命令は、プログラム、工程、サブルーチン、関数などを含むタスクのセットにグループ化される。分離されたローカルメモリ240に関連付けられた処理回路245は、本発明の原理にしたがって通信ネットワーク100を通って、改訂を蓄積されたデータおよび命令に伝播するために、その内部に記憶された命令の選択された1つを実行するように動作する。

【0026】好ましい実施の形態において、ディスプレイ210は、複数の命令の1つを実行するためにアクセスされる表示領域250を提供するように動作し、またグラフィックユーザインターフェースを表示することができる。シャーシ205上にはさらに個々の従来のコネクタ（図示せず）が接続されている。シャーシ205上の個々の従来のコネクタ（図示せず）にはマウス225とプリンタ260がさらに接続されている。処理回路245と関連付けされた、周辺機器210、215、255および260によりユーザは処理システム200と対話できるようになる。

【0027】周辺装置210、215、255および260は他の従来のインターフェースに置き換えまたは組み合わせることもできる。処理システム200は単一のプロセッサ、単一のハードディスクドライブおよび単一のローカルメモリを有するものとして例示したが、処理システム200は多重のプロセッサあるいは記憶装置の組み合わせで構成することもできる。処理システム200は、実際には、本発明の原理にしたがって、高性能計

算機、およびポータル、ラップトップ／ノートブック、ミニ、メインフレームおよびスーパーコンピュータ、電話システム（例えば、音声、ビデオ、データなど）、メッセージページングシステム、可搬装置など、およびこれらのネットワークを組み合わせたものの、いずれかの適当なノードに置き換えられあるいは組み合わせて使用される。

【0028】従来の処理システムのアーキテクチャについては、William StallingsによるMacMillian Publoshing Co.（3rd ed. 1993）のComputer Organization andArchitecureにより詳しく説明されており、従来の処理システムのネットワークデザインは、Darren L. SpohnによるMcGraw－Hill, Inc.（1993）のData Netword Designにより詳しく説明されている。また、従来のデータ通信については、R. D. Giltin, J. F. HayesおよびS. B. WeinstainによるPlenum Press（1992）のData CommunicationsPrinciplesおよびJames Harry GreenによるIrwin Professional Publishing（2nd ed. 1993）のThe Irwin Handbool of Telecommunicationsにより詳しく説明されている。これらの出版物を本明細書に参考として組み入れる。

【0029】図3に戻って、図3は図2のPC200のような、処理システムに関連付けられるの適した、マイクロプロセッシング回路（全体を符号300で示した）の高レベルのブロックダイヤグラムを例示したものである。マイクロプロセッシング回路300は、分離されたローカルメモリ240、処理回路245、バス制御回路305、従来型のリードオンリーメモリ（ROM）310および周辺機器ポート315のセットを含んでいる。ホストバス320は処理回路245、分離されたローカルメモリ240およびバス制御回路305と関連して動作する。例示した実施の形態においては、分離されたローカルメモリ240は好適にはランダムアクセスメモリ（RAM）を含んでおり、また処理回路245は好適には1つまたはそれより多くの協力して動作するプロセッサを含んでいる。

【0030】入力／出力（I／O）バス325はバス制御回路305、ROM310および周辺機器ポート315のセットと関連して動作する。周辺機器ポート315のセットはI／Oバス325を図2の周辺機器210、215、255、および260と通信のために接続する。周辺機器ポート315のセットに含まれるのは、シリアルあるいはパラレルポートである。バス制御回路305は、ホストバス320およびI／Oバス325が関

連付けされる適当な手段を提供しており、これにより、これらの間での通信のための経路と管理が提供されている。例示した実施の形態において、ホストバス３２０は、処理回路２４５と分離されたローカルメモリ２４０との間の迅速な通信を容易化するために比較的高速なものであり、またその速度を最大とするために典型的にはできるだけ少ない構成要素だけが接続されている。Ｉ／Ｏバス３２５は、その速度はあまり重要でないので、ホストバス３２０に関してより遅いペースで動作できるようになっている。バス３２０、３２５の各ラインは、信号をその上での駆動電流を必要とする。したがって、本発明は、必要とされる駆動電流を供給する従来のシステム制御器（図示せず）と共同して動作する。当然のことであるが、本発明は単一バスのアーキテクチャにおいても好適に動作するものである。

【００３１】他の好ましい実施の形態において、種々の形式の回路を構成するため、マイクロプロセッシング回路３００は、一部ないし全部が、プログラム可能な論理配列（ＰＡＬ）のようなプログラム可能プログラム可能な論理装置、デジタル信号プロセッサ（ＤＳＰ）、フィールドプログラマブルゲートアレイ（ＦＰＧＡ）、専用集積回路（ＡＳＩＣ）、大規模集積回路（ＶＬＳＩ）などを含む適当な処理回路に置き換え、組み合わせられるものである。

【００３２】次に、図４を参照して、図４は図１の通信ネットワーク１００の、単一のブランチの高レベルブロックダイヤグラム（全体を符号４００で示した）、つまり倒壊した階層（collapsed hierarchy）を例示したものである。サーバ１１０および例示したノード内の階層的な通信経路を構成するクライアントノード１２０と１３０ａはそれぞれ従来の通信リンク４０５および４１０を経て適切に関連付けされている。図４は図５だけに関して例示目的で示したものである。例示した実施の形態はツリーを基礎として階層的ネットワークに焦点を合わせたものであるが、当業者であれば、本発明の原理をあらゆる適切に配列された通信ネットワーク環境（例えば、ピアツウピアネットワークなど）に適用できるものである。本発明は、１つまたはそれより多くの第１のレベルのノードに関連した情報への改訂が、１つまたはそれより多くの第２のレベルのノードの少なくとも１つから１つまたはそれより多くの第３のノードに、１つまたはそれより多くの第３のレベルのノードの少なくとも１つから１つまたはそれより多くの第４のノードなどに、１つまたはそれより多くの第２のレベルのノードに適切に通信される手段を提供するものである。本発明によれば、それ故、通信ネットワークを通って改訂を指数的に速度で伝播することが容易化される。

【００３３】図５を参照して、図５は、図４のブランチ４００、より詳しくは本発明によりクライアントノード１２０ａと１３０ａを通って改訂を伝播するための通信

ネットワーク１００の動作の一例の方法のフローダイヤグラムを例示したものである。この説明は図４を参照して行われるものであり、またノード１１０、１２０ａおよび１３０ａのそれぞれは、図３のマイクロプロセッシング回路３００あるいは同等な機能を提供できる他の適切な構成のような適当な処理手段を含んでいる。例示したソースコードの実施の形態は付録Ａとして添付し、参考として組み入れる。例示した実施の形態はＵＮＩＸ環境、Ｓｙｓｔｅｍ Ｖ Ｒｅｌｅａｓｅ４（ＳＶＲ４）で使用される従来のＫｏｒｎ Ｓｈｅｌｌコードで書かれている。

【００３４】また、サーバノード１１０はそれに関連したメモリ内に蓄積された情報への改訂を受信する。改訂は、ソフトウェアプロバイダ／ベンダーを含む、多くのソースのいずれかから適切に受信される。

【００３５】例示したプロセスは、クライアントノード１２０ａがその関連するメモリについてその上に蓄積された情報（例えば、ファイル、データベース、データ構造、プログラム、ルーチン、サブルーチン、関数、タスクなど）を走査することで開始され、状態報告を発生する（プロセスステップ５００）。状態報告はクライアントノード１２０ａの現在の状態を示し、種々のクライアントノードの情報、種々のクライアントノードの情報に関連したバージョン番号、種々のクライアントノードの情報に関連した改訂日などを識別する識別子を含んでいる。走査プロセスはサーバノード１１０により外部から、あるいはクライアントノード１２０ａにより内部から開始される。いずれの場合でも、開始は周期的または無周期的に行われる。

【００３６】周辺機器ポート３１５の１つを使用する、クライアントノード１２０ａは、状態報告をサーバノード１１０に送信する（入力／出力ステップ５０５）。サーバノード１１０は、その周辺機器ポート３１５の１つを使用し、送信された状態報告を受信し、また送信の精度を検証する（プロセスステップ５１０）。データの送信を検証する技術は公知である。

【００３７】好ましい実施の形態では、サーバノード１１０はさらに、クライアントノード１２０ａ上でロギングすることで、クライアントノード１２０ａの現在の状態を認証するために動作し、受信された状態報告内の情報を確認する。「認証」は、本明細書においては、確認、確証、証明、実証、立証、照合などを含む、本物であることの信用を確保することを意味する。例えば、受信した状態報告がクライアントノード１２０ａがソフトウェアパッケージＸＹＺのバージョン１．０を含むことを示した場合、サーバ１１０は、クライアントノード１２０ａが実際にソフトウェアパッケージＸＹＺのバージョン１．０を含むことを確認するためにクライアントノード１２０ａにログをする。しばしば、ソフトウェアパッケージＸＹＺのような、蓄積された情報は、複数のフ

ァイルを含んでいる。関連した実施の形態では、サーバ１１０はファイル毎に状態報告を認証する。

【００３８】サーバノード１１０は、状態報告を正しく受信した場合、受信した状態報告を関数として、情報を蓄積したクライアントノード１２０ａの改訂が必要かどうかを決定する（決定ステップ５１５）。他の好ましい実施の形態においては、サーバノード１１０により、直接または間接的に、適当な登録簿（inventory ）が維持される。登録簿は、サーバノード１１０、およびクライアントノード１２０ａなどにより維持され、使用され、提供されるなどの情報のリストを含んでいる。情報を蓄積したクライアントノード１２０ａが改訂を必要とするかどうかの決定は、登録簿と状態報告を比較することで行われ、これにより、（１）クライアントノード１２０ａから失われ、（２）クライアントノード１２０ａから除かれた、（３）最近のバージョンではない、（４）実施許諾契約下で期限満了した、などの情報が識別される。比較を行うための従来技術は公知である。

【００３９】実施許諾契約に関連して、上記の識別プロセスは、クライアントノード１２０ａにより維持された、有効、無効、期限経過などの加入者情報を識別するために使用され、本発明の特徴は以下により詳細に説明する。

【００４０】情報が蓄積されたクライアントノード１２０ａが改訂を必要とする場合（決定ブロック５１５のＹＥＳ分岐）、サーバノード１１０はクライアントノード１２０ａに送信するための情報改訂ファイルを生成する（プロセスステップ５２０）。例示的な情報改訂ファイルは、プログラム、関数、タスク、サブルーチン、工程、ドキュメント、スプレッドシート、データベース、データ構造などを含んでる。改訂ファイルはまた、クライアントノード１２０ａにより実行される命令のセットを含んでおり、命令の実行されるセットはクライアントノード１２０ａに情報改訂ファイルの残りをインストールし、送信検証、機密保護などを行うように指示する。

【００４１】サーバノード１１０は、周辺装置ポート３１５の１つを使用し、クライアントノード１２０ａに改訂ファイルを送信する（入力／出力ステップ５２５）。クライアントノード１２０ａは、周辺機器ポート３１５の１つを再度使用し、送信された改訂ファイルを受信し、また送信の精度を検証する（プロセスステップ５３０）。送信が正しく受信された場合、クライアントノード１２０ａ上で蓄積された情報は受信されたバージョンファイルを使用して更新される（プロセスステップ５３５）。

【００４２】上記した更新はいろいろな方法で行うことができる。例えば、サーバノード１１０はクライアントノード１２０ａ上でロギングにより情報が蓄積されたクライアントノード１２０ａを更新し、また（ａ）従来の「マスターースレーブ」タイプの環境で更新がおこなわれ（つまり、マスタと称される側がセッションを開始し制御し、スレーブと称される他側がマスタのコマンドに応答する通信セッション、）、および（ｂ）クライアントノード１２０ａにコマンドのシーケンスを送信し、クライアントノード１２０ａによる実行により、クライアントノード１２０ａに更新をさせる。

【００４３】他の例では、クライアントノード１２０ａはサーバ１１０から改訂ファイルを受信し、受信した改訂ファイルを緩衝記憶し、適切に更新を実行する。クライアントノード１２０ａもまた改訂ファイルの一部として上記した命令のセットを受信し、命令のセットは、クライアントノード１２０ａにより実行されたときには、クライアントノード１２０ａは、緩衝記憶された改訂ファイルの残りをインストールし、あるいは送信検証、機密保持などをするように指示する。

【００４４】例示した実施の形態においては、ネットワークブランチ４００内のクライアントノード１２０ａ、第２のレベルのクライアントは、ネットワークブランチ４００内のクライアントノード１３０ａ、第３のレベルのクライアントに対して、一時的な「サーバ」として機能する。

【００４５】例示したプロセスは、クライアントノード１３０ａがその関連したメモリを走査したときに継続され、それに記憶された情報（例えば、ファイル、データベース、データ構造、プログラム、ルーチン、サブルーチン、関数、タスクなど）を学習し、また状態報告を発生する（プロセスステップ５４０）。状態報告はクライアントノード１３０ａの情報の現在の状態を表し、種々のクライアントノードの情報、種々のクライアントノードの情報に関連したバージョン番号、種々のクライアントノードの情報に関連した改訂日などを識別する識別子を含んでいる。走査プロセスは、好ましくはクライアントノード１２０ａにより外部から、あるいはクライアントノード１３０ａにより内部から開始される。いずれの場合でも、開始は周期的または無周期的に行われる。

【００４６】クライアントノード１３０ａは、周辺機器ポート３１５の１つを使用して、クライアントノード１２０ａに状態報告を送信する（入力／出力ステップ５４５）。クライアントノード１２０ａは、その周辺機器ポート３１５の１つを使用して、送信された状態報告を受信し、また送信の精度を検証する（プロセスステップ５５０）。

【００４７】好ましい実施の形態においては、クライアントノード１２０ａはさらに、クライアントノード１３０ａ上でロギングによりクライアントノード１３０ａの現在の状態を認証し、また受信した状態報告内の情報を確認する。関連した実施の形態において、その一部に複数のファイルを含む、状態報告の認証は、ファイル毎に行われる。

【００４８】クライアントノード１２０ａは、状態報告

ファイルを含んでいる。関連した実施の形態では、サーバ１１０はファイル毎に状態報告を認証する。

【００３８】サーバノード１１０は、状態報告を正しく受信した場合、受信した状態報告を関数として、情報を蓄積したクライアントノード１２０ａの改訂が必要かどうかを決定する（決定ステップ５１５）。他の好ましい実施の形態においては、サーバノード１１０により、直接または間接的に、適当な登録簿（inventory ）が維持される。登録簿は、サーバノード１１０、およびクライアントノード１２０ａなどにより維持され、使用され、提供されるなどの情報のリストを含んでいる。情報を蓄積したクライアントノード１２０ａが改訂を必要とするかどうかの決定は、登録簿と状態報告を比較することで行われ、これにより、（１）クライアントノード１２０ａから失われ、（２）クライアントノード１２０ａから除かれた、（３）最近のバージョンではない、（４）実施許諾契約下で期限満了した、などの情報が識別される。比較を行うための従来技術は公知である。

【００３９】実施許諾契約に関連して、上記の識別プロセスは、クライアントノード１２０ａにより維持された、有効、無効、期限経過などの加入者情報を識別するために使用され、本発明の特徴は以下により詳細に説明する。

【００４０】情報が蓄積されたクライアントノード１２０ａが改訂を必要とする場合（決定ブロック５１５のＹＥＳ分岐）、サーバノード１１０はクライアントノード１２０ａに送信するための情報改訂ファイルを生成する（プロセスステップ５２０）。例示的な情報改訂ファイルは、プログラム、関数、タスク、サブルーチン、工程、ドキュメント、スプレッドシート、データベース、データ構造などを含んでる。改訂ファイルはまた、クライアントノード１２０ａにより実行される命令のセットを含んでおり、命令の実行されるセットはクライアントノード１２０ａに情報改訂ファイルの残りをインストールし、送信検証、機密保護などを行うように指示する。

【００４１】サーバノード１１０は、周辺装置ポート３１５の１つを使用し、クライアントノード１２０ａに改訂ファイルを送信する（入力／出力ステップ５２５）。クライアントノード１２０ａは、周辺機器ポート３１５の１つを再度使用し、送信された改訂ファイルを受信し、また送信の精度を検証する（プロセスステップ５３０）。送信が正しく受信された場合、クライアントノード１２０ａ上で蓄積された情報は受信されたバージョンファイルを使用して更新される（プロセスステップ５３５）。

【００４２】上記した更新はいろいろな方法で行うことができる。例えば、サーバノード１１０はクライアントノード１２０ａ上でロギングにより情報が蓄積されたクライアントノード１２０ａを更新し、また（ａ）従来の「マスター－スレーブ」タイプの環境で更新がおこなわれ（つまり、マスタと称される側がセッションを開始し制御し、スレーブと称される他側がマスタのコマンドに応答する通信セッション、）、および（ｂ）クライアントノード１２０ａにコマンドのシーケンスを送信し、クライアントノード１２０ａによる実行により、クライアントノード１２０ａに更新をさせる。

【００４３】他の例では、クライアントノード１２０ａはサーバ１１０から改訂ファイルを受信し、受信した改訂ファイルを緩衝記憶し、適切に更新を実行する。クライアントノード１２０ａもまた改訂ファイルの一部として上記した命令のセットを受信し、命令のセットは、クライアントノード１２０ａにより実行されたときには、クライアントノード１２０ａは、緩衝記憶された改訂ファイルの残りをインストールし、あるいは送信検証、機密保持などをするように指示する。

【００４４】例示した実施の形態においては、ネットワークブランチ４００内のクライアントノード１２０ａ、第２のレベルのクライアントは、ネットワークブランチ４００内のクライアントノード１３０ａ、第３のレベルのクライアントに対して、一時的な「サーバ」として機能する。

【００４５】例示したプロセスは、クライアントノード１３０ａがその関連したメモリを走査したときに継続され、それに記憶された情報（例えば、ファイル、データベース、データ構造、プログラム、ルーチン、サブルーチン、関数、タスクなど）を学習し、また状態報告を発生する（プロセスステップ５４０）。状態報告はクライアントノード１３０ａの情報の現在の状態を表し、種々のクライアントノードの情報、種々のクライアントノードの情報に関連したバージョン番号、種々のクライアントノードの情報に関連した改訂日などを識別する識別子を含んでいる。走査プロセスは、好ましくはクライアントノード１２０ａにより外部から、あるいはクライアントノード１３０ａにより内部から開始される。いずれの場合でも、開始は周期的または無周期的に行われる。

【００４６】クライアントノード１３０ａは、周辺機器ポート３１５の１つを使用して、クライアントノード１２０ａに状態報告を送信する（入力／出力ステップ５４５）。クライアントノード１２０ａは、その周辺機器ポート３１５の１つを使用して、送信された状態報告を受信し、また送信の精度を検証する（プロセスステップ５５０）。

【００４７】好ましい実施の形態においては、クライアントノード１２０ａはさらに、クライアントノード１３０ａ上でロギングによりクライアントノード１３０ａの現在の状態を認証し、また受信した状態報告内の情報を確認する。関連した実施の形態において、その一部に複数のファイルを含む、状態報告の認証は、ファイル毎に行われる。

【００４８】クライアントノード１２０ａは、状態報告

が正しく受信された場合、受信した状態報告を関数として、情報が蓄積されたクライアントノード１３０ａの改訂が必要かどうかを決定する（決定ステップ５５５）。さらに別の実施の形態においては、クライアントノード１２０ａにより、直接的または間接的に、適当な登録簿が維持される。この登録簿は、クライアントノード１２０ａ、およびクライアントノード１３０ａなどにより維持され、使用され、提供されるなどの情報のリストを含んでいる。情報を蓄積したクライアントノード１３０ａが改訂を必要とするかどうかの決定は、登録簿と状態報告を比較することで行われ、これにより、（１）クライアントノード１３０ａから失われ、（２）クライアントノード１３０ａから除かれた、（３）最近のバージョンではない、（４）実施許諾契約下で期限満了した、などの情報が識別される。比較を行うための従来技術は公知である。

【００４９】実施許諾契約に関連して、上記の識別プロセスは、クライアントノード１３０ａにより維持された、有効、無効、期限経過などの加入者情報を識別するために使用され、本発明の特徴は以下により詳細に説明する。情報が蓄積されたクライアントノード１３０ａが改訂を必要とする場合（決定ブロック５５５のＹＥＳ分岐）、クライアントノード１２０ａはクライアントノード１３０ａに送信するための情報改訂ファイルを生成する（プロセスステップ５６０）。情報改訂ファイルは、クライアントノード１２０ａによりサーバノード１１０から受信した改訂ファイルの少なくとも一部を含んでいる。

【００５０】例示的な情報改訂ファイルは、プログラム、関数、タスク、サブルーチン、工程、ドキュメント、スプレッドシート、データベース、データ構造などを含んでる。改訂ファイルはまた、クライアントノード１３０ａにより実行される命令のセットを含んでおり、命令の実行されるセットはクライアントノード１３０ａに情報改訂ファイルの残りをインストールし、送信検証、機密保護などを行うように指示する。

【００５１】クライアントノード１２０ａは、周辺装置ポート３１５の１つを使用し、クライアントノード１３０ａに改訂ファイルを送信する（入力／出力ステップ５６５）。クライアントノード１３０ａは、その周辺機器ポート３１５の１つを再度使用し、送信された改訂ファイルを受信し、また送信の精度を検証する（プロセスステップ５７０）。送信が正しく受信された場合、クライアントノード１３０ａ上で蓄積された情報は受信されたバージョンファイルを使用して更新される（プロセスステップ５７５）。

【００５２】上記した更新はいろいろな方法で行うことができる。例えば、クライアントノード１２０ａはクライアントノード１３０ａ上でロギングにより情報が蓄積されたクライアントノード１３０ａを更新し、また

（ａ）従来の「マスター－スレーブ」タイプの環境で更新がおこなわれ（つまり、マスタと称される側がセッションを開始し制御し、スレーブと称される他側がマスタのコマンドに応答する通信セッション、）、および

（ｂ）クライアントノード１３０ａにコマンドのシーケンスを送信し、クライアントノード１３０ａによる実行により、クライアントノード１３０ａに更新をさせる。

【００５３】他の例では、クライアントノード１３０ａはクライアントノード１２０ａから改訂ファイルを受信し、受信した改訂ファイルを緩衝記憶し、適切に更新を実行する。クライアントノード１３０ａもまた改訂ファイルの一部として上記した命令のセットを受信し、命令のセットは、クライアントノード１３０ａにより実行されたときには、クライアントノード１３０ａは、緩衝記憶された改訂ファイルの残りをインストールし、あるいは送信検証、機密保持などをするように指示する。

【００５４】上記で説明したプロセスは、従来のネットワーク環境において適切に実行され、また下層のネットワークオペレーティングシステム（ＮＯＳ）に完全に透過的である。この特徴によって、従来の機密保持およびＮＯＳの他の特徴をそのまま利用できる。

【００５５】本発明の種々の実施の形態の重要な特徴は、クライアントノード１２０ａあるいは１３０ａのいずれかが、それ自体が受信された情報のバージョンファイルにより更新される、上記した更新プロセスの少なくとも一部を実行するための命令のシーケンスを含んでいることである。命令のシーケンスは、受信した改訂ファイルにより改訂され、これにより、クライアントノード１２０ａあるいは１３０ａの１つにおける１つまたはそれより多くの動作が変更され、また常時変化あるいは更新できるようになる。

【００５６】本発明の他の特徴は例示した実施の形態により示したように、情報を蓄積したクライアントノード１３０ａが、少なくとも一部が、サーバノード１１０からクライアントノード１２０ａにより受信された改訂ファイルにより、改訂されることである。これにより、改訂は、通信ネットワークを通り、その第１、第２および第３のノードを経て伝播される。

【００５７】関連する実施の形態においては、クライアントノード１２０ａにより発生された状態報告はクライアントノード１２０ａからサーバノード１１０に第１の時に送信され、クライアントノード１３０ａにより発生された状態報告は次いでクライアントノード１３０ａからクライアントノード１２０ａに第２の時に送信される。第２の時間は、第１の時間から、クライアントノード１２０ａが改訂をクライアントノード１３０ａに送信する前にクライアントノード１２０ａが情報を全て改訂するのに十分な時間時間の後である。本発明の別の特徴はよって、ネットワークを通っての改訂の「波」が順序付けされることである。他の実施の形態では、改訂は、

1つのノードが他のものの改訂の必要を決定するように、よりランダムな態様で分布される。

【0058】本発明の好ましい用途は、加入者ベースのソフトウェア分配システムである。「加入者ベースのシステム」は、本明細書においては、当事者である「加入者」がベンダー、ディストリビュータ、ライセンサなどと特定のソフトウェアパッケージ、ソフトウェアパッケージのグループ、電子バービスなどの特定数の発行やバージョンなどの受領および支払を契約する、電子通信システムを意味する。より詳しくは、少なくとも1つのサーバノード110あるいはクライアントノード120aは、加入者リストを蓄積するためのメモリを含み、加入者に対して加入したサービスと関係付けられている。サーバノード110およびクライアントノード120aは改訂ファイルの、少なくと一部を、加入者リストのコンテントの関数として送信する。本発明は、それ故、加入者が改訂に対して支払をする、料金ベースの更新のコアを形成するものである。改訂される情報の量および改訂の頻度は選択可能であり、広い範囲の料金ベースのサービスが提供される。関連した実施の形態では、加入者リストは部外秘リストに関連している。部外秘リストは、加入者リストと関連して処理された場合には、例えば、地理的な位置により関連したユーザグループのような、ユーザの特定のユーザまたはグループに利用可能あるいは利用可能でない加入者リストの情報のサブセットを識別する。部外秘リストはよって、加入者リストに対するフィルタとして機能する。

【0059】インターネットのようなWANだけでなく、ビデオゲームや他の対話型のサービスを受信するためのサービスを含むペイパービュー方式およびデマンド方式のテレビを提供するケーブルテレビによる、加入者ベースのシステムを通して更新を伝播すること特に好ましい。

【0060】上記のように、本発明によれば、複数の関連付けされたノードを含む、通信ネットワークを通して改訂を伝播するためのシステム、および方法が提供される。本発明のシステムは、（1）第2のノードのメモリに蓄積された第2のノード情報の現在の状態を収集し送信するために、第2の通信ネットワークの第2のノードに関連付けされた状態報告回路、（2）第2のノードから現在の状態を受信し、現在の状態の関数として第2のノード情報の改訂が必要かどうかを決定し、また改訂が必要である場合には、第2のノード情報を改訂するために改訂を第2のノードに送信するために、通信ネットワークの第1のノードに関連付けされた第1の情報改訂回路、（3）通信ネットワークの第3のノードから現在の状態を受信し、第3のノードから前記現在の状態の関数として第3のノードのメモリ内に蓄積された第3のノードの改訂が必要かどうかを決定し、改訂が必要である場合に、第3のノード情報を改訂するために第2のノードから受信した改訂を第3の第3のノードに送信し、改訂が第1、第2、および第3ノードを経由し通信ネットワークを通って伝播される、通信ネットワークの前記第2のノードに関連付けされた第2の情報改訂回路、を含んでいる。よって、改訂は自動的に、通信ネットワークを通って伝播され、通信ネットワーク内の種々のノードは他のノードにおける情報の改訂が必要なときを検出することと、他のノードに改訂を送信することの責務がある。

【0061】本発明の範囲は、図1、図4および図5に示した、ツリーベースの階層的ネットワークに限定されず、ピアツウピア通信ネットワークのような他の従来のネットワーク構造を含むものである。同様に、改訂の伝播は第1のノード、第2のノード、および第3のノードへに限定されず、第1のノードから1つまたはそれより多くの第2のノードへ、少なくとも1つの1つまたはそれより多くの第2のノードから1つまたはそれより多くの第3のノードへ、少なくとも1つの1つまたはそれより多くの第3のノードから1つまたはそれより多くの第4のノードなどを含み、直列的に関連したノードの順次的な改訂だけでなく、複数のノードの階層的なファンアウトの更新を含むものである。

【0062】さらに、本発明は、LANやWANなどん純粋な「コンピュータベース」の通信ネットワークに限定されず、セルラー電話あるいはメッセージページング値とワークのような無線環境におけるシステムソフトウェアあるいはデータを更新する通信システムにも適用できるものである。このため、本発明の原理は、ルータ、ブリッジ、ゲートウェイ、交換機あるいは他の可搬性の装置、人工衛星、リレー局などを含む、ノードとして機能するいずれかのネットワーク要素に関連している。以上、本発明の原理を説明したが、当業者には本発明の技術と範囲を逸脱することなしに種々の変更、置き換え、および交換を容易になし得るものである。

【0063】

21 22

付 録 A

「ＳＥＮＤＬＩＳＴ」のソースコード

```
CMD=`basename $0`
USAGE="Usage: $CMD { -d } [ -s serverID ]"

KSHOK=no
echo yes | read KSHOK
if test "$KSHOK" = "no"; then
  if test "${RETRYING_KSH:-no}" = "yes"; then
        echo "$CMD: ERROR: not running with ksh88 -- aborting!"
        echo "$CMD: ERROR: not running with ksh88 -- aborting!" |
        /bin/mail exptools
        RC=2
  elif test $# -gt 0; then
        RETRYING_KSH="yes" $SHELL $0 "$@"
        RC=$?
  else
        RETRYING_KSH="yes" $SHELL $0
        RC=$?
  fi
  exit $RC
fi


# ---------------------< emsg
#
# 本ルーチンはエラーメッセージのプリントアウトおよびｅｘｐｔｏｏｌへの
  通知のためのものである
emsg() {
  typeset MSG="$CMD: ERROR: $1 -- aborting!"
  typeset LOG=$ADMRUG/$SERVERID/local/sendplist.out
  typeset ECODE ECMD

  if test -n "$2"; then
        echo $2 | read ECMD ECODE
        MSG="$MSG\nError '$ECODE' reported by '$ECMD'"
  fi

  echo "$MSG" >&2
  {
        echo "Subject: sendplist error!"
        echo
        echo "$MSG"
        if test -s $LOG; then
          echo "\nHere is the complete sendplist log file:"
          echo "-------------------------------------------------------"
          pr -o4 -t $LOG
          echo "-------------------------------------------------------"
        fi
  } | /bin/mail exptools
}


# ---------------------< extractsection
#
# 本ルーチンはＥＯＦラインにより終了した入力ファイルの所定のセクションを
  抽出するためのものである
```

【００６４】

23 24

```
extractsection() {
   integer section=$1
   typeset inputfile=$2
   integer i=1
   typeset LINE done=false

   while not $done && read LINE; do
         if ((i > section)); then
            done=true
         elif test "$LINE" = "$EOF"; then
            ((i += 1))
         elif ((i == section)); then
            echo "$LINE"
         fi
   done < $inputfile
}

# ---------------------------< main


GETOPT=$(getopt ds: "$@")
if (($? != 0)); then
   echo "$USAGE"
   exit 2
fi

set -- $GETOPT

#set -e                                  # エラーの場合には出る

debug=""
SERVERID=""
for arg in "$@"; do
   case "$arg" in
   -s)
         SERVERID=$2
         shift 2
   ;;
   -d)
         debug=:
         shift
   ;;
   --)
         shift
         break
   ;;
   esac
done

if test ! -f $ADMRUG/global/config; then
   emsg "Can't find RUG global config file"
   exit 2
fi

. $ADMRUG/global/config

if test -n "$SERVERID"; then
   FOUND=false
```

【0065】

```
    25                                                          26
  for ID in $SERVERIDLIST; do
        if test "$ID" = "$SERVERID"; then
          FOUND=true
        fi
  done
  if not $FOUND; then
        emsg "Unknown serverid '$SERVERID' given"
        exit 2
  fi
else
  if test -z "$DEFAULTSERVERID"; then
        emsg "Default server id missing"
        exit 2
  fi
  SERVERID=$DEFAULTSERVERID
fi

exec > $ADMRUG/$SERVERID/local/sendplist.out 2>&1

echo "Start `date`\n"

CASCADEFILE=$ADMRUG/$SERVERID/global/cascade
if test -f $CASCADEFILE; then
   cp $CASCADEFILE $ADMRUG/local/cascadefile
   chmod 664 $ADMRUG/local/cascadefile
fi

. $ADMRUG/global/linkconfig $SERVERID

NETINFO=$(echo $SERVERID $TYPE | $NETCMD)
if test -z "$NETINFO"; then
   emsg "Unable to acquire network information"
   exit 2
fi

echo "Networking information: $NETINFO"

tmppkglist=/usr/tmp/$$pkglist
tmpplist=/usr/tmp/$$plist
tmpexclude=/usr/tmp/$$exclude
tmpignore=/usr/tmp/$$ignore
mkperrs=/usr/tmp/$$mkperrs
errfile=/usr/tmp/$$errfile
tmpsubfile=/usr/tmp/$$subfile
tmpflist="$tmppkglist $tmpplist $tmpexclude $tmpignore $mkperrs $errfile $tmpsubfile"

retval=0
trap '
   retval=1
   exit
' 1 2 3 15
trap '
   rm -f $tmpflist
   exit $retval
' EXIT

cd

rm -f adm/upd1.1/lib/cpio.new                    # 緊急の更新リストを除去する

echo "\nComputing the subscription list ..."
```

【0066】

27 28

```
{
  echo "The $LOGNAME userid uses the alias '$LOCALCLIENTID' to subscribe to the
following $TYPE tools from '$SERVERID':"
  cat $SUBSCRLIST |                          # 全てのｓｕｂｓｃｒリストファイルを結合する
  CleanComm |
  sed '/^!/d' |
  sort -f > $tmpsubfile
  if test -s $tmpsubfile; then
        pr -o4 -t -4 $tmpsubfile
  else
        echo "NONE!"
  fi
  cat $SUBSCRLIST |                          # 全てのｓｕｂｓｃｒリストファイルを結合する
  CleanComm |
  sed '/^[^!]/d; s/^!//' |
  sort -f > $tmpsubfile
  if test -s $tmpsubfile; then
        echo "  rejecting these tools:"
        pr -o4 -t -4 $tmpsubfile
  fi
} > $ADMRUG/local/subscrlist
chmod 664 $ADMRUG/local/subscrlist

echo "\nComputing checksums..."

(
  > $errfile
  > $mkperrs
  {
        cat $EXCLUDELIST                     # 全てのｅｘｃｌｕｄｅリストファイルを結合する
        genlinkdirs $SERVERID
  } | mksed > > $tmpignore
  { find . ! -name "." -print       || echo "find $?" > $errfile ;} |
  { sed -e 's!^\./!!' -f $tmpignore || echo "sed $?" > $errfile ;} |
  { $TADMRUG/bin/mkplist -m 2 > $mkperrs ||
                                        echo "mkplist $?" > $errfile ;} |
  { sort -u                          || echo "sort $?" > $errfile ;}
  if test -s $errfile; then
        if test -s $mkperrs; then
          cat $mkperrs > > $ADMRUG/$SERVERID/local/sendplist.out
        fi
        emsg "Failure preparing client plist report" "$(< $errfile)"
        exit 2
  else
        echo "$EOF"
  fi
) > $tmpplist
chmod 664 $tmpplist

eval ${debug:+"cp $tmpplist $ADMRUG/$SERVERID"}

if ((${TZ=GMT date +%H} < 2)); then          # ２ＡＭ　ＧＭＴの以前の場合
  integer WAITTIME=RANDOM%1800               # １／２時間のランダムな遅延をセット
  echo "\nWaiting $WAITTIME seconds to prevent server overload"
  sleep $WAITTIME
fi

echo "\nSending to $SERVERID"

echo "$NETINFO" |
$debug $SENDCMD -u $SERVERID_LOGNAME -f rje/$SERVERID/$LOCALCLIENTID.$YRDAY $tmpplist
```

【００６７】

```
(
  > $errfile                                  # 全てのｓｕｂｓｃｒリストファイルを結合する
  cat $SUBSCRLIST |
  CleanComm |
  sort -u
  echo "$EOF"
  (
        cat $EXCLUDELIST                      # 全てのｅｘｃｌｕｄｅリストファイルを結合する
        genlinkdirs $SERVERID
  ) |
  CleanComm |
  sort -u
  echo "$EOF"
  if test -z "$DEFAULTLOCALID"; then
        echo "0:$LOCALCLIENTID:$(uname -n):$LOGNAME:$(uname -rvm)"
  else
        typeset LINKDIR=$ADMRUG/$DEFAULTLOCALID
        typeset RJE=$HOME/rje/$DEFAULTLOCALID
        echo "0:$LOCALCLIENTID/$DEFAULTLOCALID:$(uname -n):$LOGNAME:$(uname -rvm)"
        if test -f $LINKDIR/server/exptab; then
           sort $LINKDIR/server/exptab
        fi |
        while IFS=: read CLIENT REST; do
           if test -f $RJE/$CLIENT.pkg; then
                integer COUNT=0
                extractsection 3 $RJE/$CLIENT.pkg |
                while IFS=: read COUNT CT ND UID OS; do
                  ((COUNT += 1))
                  echo "$COUNT:$CT:$ND:$UID:$OS"
                done
           fi
        done
  fi
  echo "$EOF"
) > $mppkglist
chmod 664 $mppkglist

eval ${debug:+"cp $mppkglist $ADMRUG/$SERVERID"}

echo "$NETINFO" |
$debug $SENDCMD -u $SERVERID_LOGNAME -f rje/$SERVERID/$LOCALCLIENTID.pkg $mppkglist

if [ -s $mkperrs ]; then
  {
        echo "Subject: mkplist errors on $LOCALCLIENTID\n"
        sed 's/^/$LOCALCLIENTID:sendplist:$TYPE:/' $mkperrs
  } |
  /bin/mail $SERVERMAIL

  # エラーメッセージをログする
  echo "Errors:"
  cat $mkperrs
fi

rm -f /tmp/sh$$.1

echo "\nFinish `date`"
```

【００６８】

31 32

「ＳＥＮＤＵＰＤＴ」ソースコード

```
CMD=`basename $0`
USAGE="Usage: $CMD [ -cdprR ] [ -l localID ] [ -t threshold ] client"

KSHOK=no
echo yes | read KSHOK
if test "$KSHOK" = "no"; then
  if test "${RETRYING_KSH:-no}" = "yes"; then
        echo "$CMD: ERROR: Not running with ksh88 -- aborting!"
        RC=2
  elif test $# -gt 0; then
        RETRYING_KSH="yes" $SHELL $0 "$@"
        RC=$?
  else
        RETRYING_KSH="yes" $SHELL $0
        RC=$?
  fi
  exit $RC
fi

alias -x echo="print -"



#-------------------------< emsg
#
# 本ルーチンはエラーメッセージのプリントアウトおよびｅｘｐｔｏｏｌへの
  通知のためのものである
emsg0 {
  typeset MSG="$CMD: ERROR: $1 -- aborting!"
  typeset LOG=$outfile
  typeset ECODE ECMD

  if test -n "$2"; then
        echo $2 | read ECMD ECODE
        MSG="$MSG\nError '$ECODE' reported by '$ECMD'"
  fi

  echo "$MSG" >&2
  {
        echo "Subject: sendupdates error!"
        echo
        echo "$MSG"
        if test -s $LOG; then
          echo "\nHere is the end of the sendupdate log file:"
          echo "------------------------------------------------"
          tail $LOG | pr -o4 -t
          echo "------------------------------------------------"
        fi
  } | /bin/mail exptools
}


#-------------------------< findclient
#
# 本ルーチンは最初のａｒｇにより定義されるマシンのタイアのリストに戻るものである

findclient() {
```

【００６９】

33

```
    machines type==$1 | sort
}

# ----------------------< cleanupclient
#
# 本ルーチンは目的のクライアントに対して旧ｒｅｊファイルを移すためのものである

cleanupclient() {
    typeset CLIENTID=$1 today=$2
    typeset rmpattern todayfile

    yr=`date '+%y'`
    lastyr=`expr $yr - 1`
    ${debug} rm -f $RJE/$CLIENTID.${lastyr}* $RJE/$CLIENTID.diff*
    rmpattern="$RJE/$CLIENTID.${yr}*"
    todayfile="$RJE/$CLIENTID.$today"
    for x in $rmpattern; do
        if test "$x" != "$todayfile" -a "$x" != "$rmpattern"; then
            ${debug} rm $x
        fi
    done
}

# ----------------------< deletefiles
#
# 本コマンドはクライアントからファイルを削除するためのものである

deletefiles() {
    $debug delfile -v -p $EXPPKGID -c -P $EXPPWD -M $SERVERMAIL "$@" name==$CLIENTID
}

# ----------------------< deletedirs
#
# 本コマンドはクライアントからディレクトリを削除するためのものである

deletedirs() {
    $debug rmdir -v -p $EXPPKGID -c -P $EXPPWD $aiuid -M $SERVERMAIL "$@" name==$CLIENTID
}

# ----------------------< senddelfiles
#
# 本ルーチンはクライアントからのファイルを削除するコマンドを送信するためのものである

senddelfiles() {
    typeset CLIENTID=$1 diffile=$2
    typeset pattern='^- \(.*[^/]\)$'        # 通常の（ｄｉｒではない）ファイルパターン

    if (( $(grep -c "$pattern" $diffile) > 0)); then
        echo "\tRemove files:"
        sed -n "s!$pattern!                    \1!p" $diffile
        sed -n "s!$pattern!rm '\1'!p" $diffile |
        {
            # 標準のセットアップをする
            echo 'HOME=`expr "$0" : "\(.*\)/adm/bin/"`'
            echo '. $HOME/.crosprofile'
            # ユーザのコマンドを得る
            cat
```

【００７０】

35 36

```
        } > $tmpcmdfile
        rex -cv -p $EXPPKGID -r priv -P $EXPPWD $altuid -M $SERVERMAIL -f $tmpcmdfile name==$CLIENTID
    fi
}

# ------------------------< sendchmods
#
# 本ルーチンはクライアントにｃｈｍｏｄコマンドを送信するためのものである

sendchmods() {
    typeset CLIENTID=$1 diffile=$2
    typeset pattern='^\([0-9]*\) \(.*\)$'   # chmods file pattern

    if (( $(grep -c "$pattern" $diffile) > 0)); then
        echo "\tChange files:"
        sed -n "s!$pattern!                \1          \2!p" $diffile
        sed -n "s!$pattern!chmod \1 '\2'!p" $diffile |
        {
            # Do the standard setup
            echo "HOME=`expr "$0" : "\(.*\)/adm/bin/"`"
            echo ". $HOME/.cronprofile"
            # ユーザのコマンドを待る
            cat
        } > $tmpcmdfile
        rex -cv -p $EXPPKGID -r priv -P $EXPPWD $altuid -M $SERVERMAIL -f $tmpcmdfile name==$CLIENTID
    fi
}

# ------------------------< sendrmdirs
#

sendrmdirs() {
    typeset CLIENTID=$1 diffile=$2
    typeset pattern='^- \(.*\)/$'          # ディレクトリパターン

    if (( $(grep -c "$pattern" $diffile) > 0)); then
        echo "\tRemove directory:"
        sed -n "s!$pattern!                \1!p" $diffile
        sed -n "s!$pattern!-d'\1'!p" $diffile |
        sort -r |
        xargs -s350 echo deletedirs |  # 必要な関数をエコー
        eval "$(cat)"                          # 構成された関数を実行
    fi
}

# ------------------------< sendrcpfiles
#

sendrcpfiles() {
    typeset CLIENTID=$1 diffile=$2
    integer onemeg=1048576  # １つのｍｅｇファイル内のバイト数
    integer bsize           # 本マシン上のブロック内のバイト数
    integer ilimit          # ブロック内のｃｐｉｏファイルの最大サイズ
    typeset CPIOFLAGS               # ｃｐｉｏに対するフラグ

    case $(machtype)
    in  ikm)    bsize=4096          # ｉｋｍが４０９６バイトブロックを使用
    ;;  pyr)    bsize=2048          # ｐｙｒが２０４８バイトブロックを使用
```


【００７１】

```
37                                                              38
        bsize=512 # ５１２バイトブロックは当用する
    esac

    ((limit = ofsmeg / bsize)) # ファイルのブロックサイズの限界と計算
    sed -n "s|^\(.*\)/\(.*\)$|\1 \2|p" $diffile > $tmpclist
    if [ -s "$tmpclist" ]; then
        integer sum=0 size
        echo "Update files:"
        sed 's/^/                    / $tmpclist
        > $tmpwtlist
        nawk 'ls -ed' < $tmpclist |
        {
          CPIOFLAGS="-oc"
          if test "$TYP" = "mip"; then
              CPIO=expcpio
          else
              CPIO=cpio
              if $SVR4; then
                CPIOFLAGS="-o -Hodc"
              fi
          fi
          while read -r size filename; do
            ((sum = sum + size))
              if [[ $sum -gt $limit && -s $tmpwtlist ]]; then
                $CPIO $CPIOFLAGS < $tmpwtlist > $tmpcpio
                    $debug sendcpio -v -p $EXPPKGED -c -P $EXPPWD $atmid -M $SERVERMAIL -f $tmpcpio
name==$CLIENTID
                ((sum = size))
                > $tmpwtlist
              fi
              echo "$filename" >> $tmpwtlist
          done
          $CPIO $CPIOFLAGS < $tmpwtlist > $tmpcpio
          $debug sendcpio -v -p $EXPPKGED -c -P $EXPPWD $atmid -M $SERVERMAIL -f $tmpcpio name==$CLIENTID
        }
    fi
}

#-------------------------< dirlog
#

dirlog() {
  typeset CLIENTID=$1 diffile=$2

  sed -n "/\/$/s%\(.*\).* date +%y%m%d' \1 %p' $diffile >> $LINKDIR/client/$CLIENTID/dirlog
}

#-------------------------< updlog
#

updlog() {
  typeset CLIENTID=$1 diffile=$2

  sed -n "/\/$/s%^.* date +%y%m%d' %p' $diffile >> $LINKDIR/client/$CLIENTID/updlog
}

#-------------------------< plusOK
#
```

【００７２】

39 40

```
plistOK() {
  typeset plistfile=$1
  typeset RC

  if test ! -f "$plistfile"; then
        RC=1
  elif test "$(tail -1 $plistfile 2>/dev/null)" != "$EOF"; then
        RC=1
  else
        RC=0
  fi

  return $RC
}

# ----------------------------< pkgfileOK
#
pkgfileOK() {
  typeset pkgfile=$1
  typeset RC

  if test ! -f "$pkgfile"; then
        RC=1
  elif test "$(tail -1 $pkgfile 2>/dev/null)" != "$EOF"; then
        RC=1
  else
        RC=0
  fi

  return $RC
}

# ----------------------------< mktoollist
#
# 本ルーチンは引数として与えられたファイル内に予約されたツールの全てのリストに
  戻るためのものである

mktoollist() {
  typeset RC

  > $errfile2                          # エラーなし状態で開始
  > $tmpsubscr2                        # 最初にツールリストなしで開始
  for subfile; do

        # コメントと空ラインを除く
        { CleanComm < $subfile || echo "CleanComm $?" > $errfile2 ;} |
        # 除外されたツール名を得る
        { sed '/^[^!]/s; s/^!//' || echo "sed $?" > $errfile2 ;} |
        # メタ名前を拡張する
        { expandtools || echo "expandtools $?" > $errfile2 ;} |
        # リストを分類
        { sort -u > $tmpexclude || echo "sort $?" > $errfile2 ;}

  if test -s $errfile2; then break; fi      # エラーの場合ループを中断

        # 先のツールリストに移動する
        { mv $tmpsubscr2 $tmpsubscr1 || echo "mv $?" > $errfile2 ;}
```

【0073】

41 42

```
if test -s $errfile2; then break; fi       # エラーの場合ループを中断

        # コメントおよび空ラインを除く
        { CleanComm < $subfile || echo "CleanComm $?" >$errfile2 ;} |
        # 予約されたツール名を得る
        { sed '/^!/d' || echo "sed $?" >$errfile2 ;} |
        # いずれかのメタ名前を拡張する
        { expandtools || echo "expandtools $?" >$errfile2 ;} |
        # 先のツールを加え分類する
        { sort -u - $tmpsubsec1 || echo "sort $?" >$errfile2 ;} |
        # 除外された名前を取り除く
        { comm -23 - $tmpexclude > $tmpsubsec2 || echo "comm $?" >$errfile2 ;}

 if test -s $errfile2; then break; fi       # エラーの場合ループを中断
 done

 if test -s $errfile2; then
        read ECMD ECODE < $errfile2
        echo "$0: Error '$ECODE' reported by '$ECMD'" >&2
        RC=$ECODE
 else
        cat $tmpsubsec2                     # 最後のリストに戻る
        RC=0
 fi

 return $RC
}

# ------------------------< extractsection
#
# 本ルーチンはＥＯＦラインにより終了した入力ファイルの所定のセクションを
  抽出するためのものである

extractsection() {
   integer section=$1
   typeset inputfile=$2
   integer i=1
   typeset LINE done=false

   while not $done && read LINE; do
          if ((i > section)); then
             done=true
          elif test "$LINE" = "$EOF"; then
             ((i += 1))
          elif ((i == section)); then
             echo "$LINE"
          fi
   done < $inputfile
}

# ------------------------< mailnews

mailnews() {
   typeset NEWSTIME=$HOME/.mailnews_time NEWSDIR=$ADMRUG/news

   if test -d $NEWSDIR  -a  -n "$*"; then
          cd $NEWSDIR
          for file in $(ls -tr); do
             if test $file -nt $NEWSTIME; then
```

【0074】

```
            echo "\nAnnouncement '$file' mailed to $*"
            {
                echo "Subject: Exptools announcement"
                echo
                cat $file
            } | /bin/mail $*
        fi
    done
    cd -
  fi
  touch $NEWSTIME
}
# ——————————————< salutation

salutation() {
  if not $SALUTATION; then
        SALUTATION=true
        e                    c                    h                    o
"\n================================================================
================"
        echo "—> To the Exptools Administrator of server $LOCALID:"
  fi
}


# ——————————————< clientsort
#
#本ルーチンはカスケード内のマシンの深度と数に基づいてクライアントを分類するための
#ものである。これにより、より深く多くのカスケードを持つクライアントが最初にツールを
#得ることでき、より早期に展開できる

clientsort() {
   typeset CL
   integer DEPTH MAXDEPTH MACHCOUNT WEIGHT
   for CL; do
        typeset PKGFILE=$HOME/tjs/$LOCALID/$CL.pkg
        MAXDEPTH=0 MACHCOUNT=0 WEIGHT=0
        if test -f $PKGFILE; then
           extractsection 3 $PKGFILE |
           while IFS=: read DEPTH REST; do
              if ((MAXDEPTH < DEPTH)); then
                 ((MAXDEPTH = DEPTH))
              fi
              if ((DEPTH == 1)); then
                 ((MACHCOUNT += 1))
              fi
           done
           ((WEIGHT = MAXDEPTH + MACHCOUNT/4))
        fi
        echo "$WEIGHT $CL"
   done |
   sort +0 -1nr +1 |
   while read DEPTH CL; do
        echo "$CL"
   done
}

# ——————————————< main
#
```

【0075】

45

```
autoload take

cd

GETOPT=$(getopt cdprRl:t: "$@")
if ((S? != 0)); then
   echo "SUSAGE"
   exit 2
fi

set -- $GETOPT

debug=""
LOCALID=""
RESEND=false
CHECKSUMS=false
USEPLIST=false
NULL_MEANS_ALL=true
: ${THRESHOLD:=300}
for arg in "$@"; do
   case "$arg" in
   -c)
         CHECKSUMS=true
         shift
   ;;
   -d)
         debug=echo
         shift
   ;;
   -r)
         RESEND=true
         NULL_MEANS_ALL=false
         shift
   ;;
   -R)
         RESEND=true
         shift
   ;;
   -p)
         USEPLIST=true
         shift
   ;;
   -t)
         THRESHOLD=$2
         shift 2
   ;;
   -l)
         LOCALID=$2
         shift 2
   ;;
   --)
         shift
         break
   ;;
   esac
done

CLIENTLIST="$@"

if test ! -f $ADMRUG/global/config; then
   echo "$CMD: ERROR: Can't find RUG global config file"
```

46

【0076】

47 48

```
    exit 2
fi

. SADMRUG/global/config

if test -n "$LOCALID"; then
    FOUND=false
    for ID in $DEFAULTLOCALID $LOCALIDLIST; do
        if test "$ID" = "$LOCALID"; then
            FOUND=true
        fi
    done
    if not $FOUND; then
        echo "$CMD: ERROR: Unknown local server ID '$LOCALID' given -- aborting!"
        exit 2
    fi
else
    if test -z "$DEFAULTLOCALID"; then
        echo "$CMD: ERROR: Default local server ID missing -- aborting!"
        exit 2
    fi
    LOCALID=$DEFAULTLOCALID
fi

. SADMRUG/global/linkconfig $LOCALID

outfile=$LINKDIR/server/update.out
if test $(find $outfile -mtime +1 -print | wc -l) -ne 0; then
    SENDUPDATES_DORMANT=true          # Ｓｅｎｄｕｐｄａｔｅｓはアスリープである
else
    SENDUPDATES_DORMANT=false         # Ｓｅｎｄｕｐｄａｔｅｓは通常に実行されている
fi
$debug eval "exec > $outfile 2>&1"
echo "Start `date`\n"

EXPTAB=$LINKDIR/server/exptab                    export EXPTAB
EXPPKGID=$LOCALID
RJE=$HOME/rje/$LOCALID
NEWFILES=$HOME/adm/upd1.1/lib/cpio.new
NEWFILESCOPY=$HOME/adm/upd1.1/lib/cpio.new2

: ${tmp:=/usr/tmp}
tmpCignore=$tmp/$$Cignore         # クライアントから無視するファイルのリスト
tmpSignore=$tmp/$$Signore # サーバから無視するファイルのリスト
tmpignfile=$tmp/$$ignfile   # 無視するワークファイルのファイル
tmpdiff=$tmp/$$pdiff              # クライアント上で変更するファイルのリスト
tmpclist=$tmp/$$clist             # クライアントへのｃｐｏｉへのファイルのリスト
tmpplist=$tmp/$$plist             # 分類された、コメントなしのクライアントのｐｌｉｓｔ
tmpwlist=$tmp/$$wlist             # ｃｐｉｏにサイズ制限されたファイルのリスト
tmpcpio=$tmp/$$cpio               # クライアントに送信するｃｐｉｏリスト
tmpexclude=$tmp/$$excl            # 除外されたツールのワークリスト
tmpsubscr1=$tmp/$$sub1            # 予約されたツールのワークリスト
tmpsubscr2=$tmp/$$sub2            # 予約されたツールのワークリスト
tmpCNTmodel=$tmp/$$Cmodel         # クライアントにより要求されたファイル名
tmpCNTplist=$tmp/$$Cplist # 除外されたファイルが取り除かれたクライアントのｐｌｉｓｔ
tmpSRVtools=$tmp/$$Stools # 提供されたツールのリスト
tmpSRVplist=$tmp/$$Splist # クライアントに提供されたファイルのｐｌｉｓｔ
tmppkgfile=$tmp/$$pkgfile   # クライアントが予約したパッケージのワークファイル
tmpfiles=$tmp/$$files             # ファイル名のワークファイル
tmpcmdfile=$tmp/$$cmds            # 送信されたコマンドのワークファイル
errfile=$tmp/$$errfile            # パイプ内のエラーを検出するためのフラグファイル
```

【００７７】

49

```
errfile2=$tmp/$$errfile                    # バイア内のエラーを捕捉するためのフラグファイル
tmpflist="$tmpCigscue $tmpSignor $tmpigsfile $tmpdiff $tmpsfax $tmpplist $tmpwfax $tmpcpio $tmpexclude $tmpCNTmodel
$tmpCNTplist $tmpSRVtools $tmpSRVplist $tmppkgfile $tmpfiles $tmpsendfile $tmpsubscr1 $tmpsubscr2 $errfile $errfile2"

retval=0
trap '
  retval=1
  exit

trap '
  trap - ERR
  xmsg "Error code $retval detected"
  exit $retval
' ERR
trap '
  $debug rm -f $tmpfiles
  {
        echo "Subject: $LOCALID gateway runlog\n"
        cat $outfile
  } |
  $debug /bin/mail $ADMIN_EMAIL
  exit $retval
' EXIT

if $SENDUPDATES_DORMANT; then
  echo "*****************************************************"
  echo "WARNING: Sendupdates has not been run for over 48 hours!\n"
  echo "*****************************************************"
fi

SRVplist=$LINKDIR/server/plist.$YRDAY
SRVflist=$LINKDIR/server/flist
for file in $LINKDIR/server/plist.*; do
  if [ "$file" != "$SRVplist" ]; then
        rm -f $file
  fi
done

if test -z "$CLIENTLIST" && $NULL_MEANS_ALL; then
  CLIENTLIST=`findclients $TYPE`
fi

echo "Clients:
$CLIENTLIST"

if $RESEND; then
  for CLIENTID in $CLIENTLIST; do
        CLIENTID=$(mkx 8 $CLIENTID)
        EXPPWD=$LINKDIR/clients/$CLIENTID/.pwd
        if test -s $LINKDIR/clients/$CLIENTID/uidsave; then
          atuid="-u$(< $LINKDIR/clients/$CLIENTID/uidsave)"
        else
          atuid=
        fi
        tmpdiff=$RJE/$CLIENTID.diff
        SUSPENDED=$LINKDIR/clients/$CLIENTID/Suspended
        if test ! -f $tmpdiff; then
          echo "\n---> No diff file found for $CLIENTID\n"
        elif test -f $SUSPENDED && $NULL_MEANS_ALL; then
          echo "\n---> Client $CLIENTID currently suspended"
        else
```

【0078】

40

50

51 52

```
        cc=`wc -l < $tmpdiff`
        if [ "$cc" -eq 0 ]; then
            echo "\n\tNo updates for $CLIENTID"
        else
            echo "\n\tChanges for $CLIENTID: " $cc
            senddelfiles $CLIENTID $tmpdiff
            sendrmdirs  $CLIENTID $tmpdiff
            sendchmods  $CLIENTID $tmpdiff
            sendrepfiles $CLIENTID $tmpdiff
            dirlog      $CLIENTID $tmpdiff
            updlog      $CLIENTID $tmpdiff
        fi
    fi
  done
elif $USEPLIST || $CHECKSUMS || not plistOK $SRVplist; then
  {
        cat $EXCLUDELIST
        genlinkdirs $LOCALID
  } | mksed -w > $tmpSignore
  if $USEPLIST; then
        echo "\nAccepting current checksums ..."
  else
        echo "\nCreating upward cascade map"
        UPDLOG=$TOOLS/adm/upd1.1/lib/updlog
        CPIOLOG=$TOOLS/adm/upd1.1/lib/cpio.log
        TZ=$OLD_TZ date | read X X X X TIMEZONE REST
        MONTH="DATE"
        DAY="UNKNOWN!"
        TIME=""
        if test -f $CPIOLOG -a ! -f $ADMRUG/.over; then
          TZ=$OLD_TZ ls -l $CPIOLOG | read X X X X X MONTH DAY TIME REST
          TIME="$TIME $TIMEZONE"
        elif test -f $UPDLOG; then
          TZ=$OLD_TZ ls -l $UPDLOG | read X X X X X MONTH DAY TIME REST
          TIME="$TIME $TIMEZONE"
        fi
        integer LEVEL
        CASCADEFILE=$LINKDIR/global/cascade
        echo "1:$LOCALID:$MONTH $DAY $TIME" > $CASCADEFILE
        for ID in $DEFAULTSERVERID $SERVERIDLIST; do
          if test -f $ADMRUG/$ID/global/cascade; then
              while IFS=: read LEVEL SERVER TIMESTAMP; do
                ((LEVEL += 1))
                echo "$LEVEL:$SERVER:$TIMESTAMP"
              done < $ADMRUG/$ID/global/cascade > > $CASCADEFILE
          fi
        done
        chmod 664 $CASCADEFILE
        cat $CASCADEFILE

        echo "\nCreating downward cascade map"
        {
          echo "\nNOTE: This cascade map generated $(date)"
          showtree -l $LOCALID
        } > $LINKDIR/local/cascade

        echo "\nChecking on expwols announcements"
        (
          LOCAL_ADMIN_EMAIL=$ADMIN_EMAIL
          if test -n "$DEFAULTSERVERID"; then
              . $ADMRUG/global/linkconfig $DEFAULTSERVERID
```

【0079】

53 54

```
        SERVER_ADMIN_EMAIL=$ADMIN_EMAIL
    else
        SERVER_ADMIN_EMAIL=""
    fi
    if test "$LOCAL_ADMIN_EMAIL" != "$SERVER_ADMIN_EMAIL"; then
        mailnews $LOCAL_ADMIN_EMAIL
    fi
)

echo "\nCalculating new checksums ..."
rm -f $RJE/UpdateStarted $RJE/UpdateEnded $RJE/RequestEnded $errfile
{
    mktoollist $SUBSCRLIST  || echo "mktoollist $?" >$errfile
} > $tmpSRVtools

if test -s $errfile; then
    cmsg "Failure preparing server tool list" "$(< $errfile)"
    rm -f $tmpSRVtools
    exit 2
fi

if test ! -s $tmpSRVtools; then
    echo "$CMD: ERROR: No tools being served by this server -- aborting!"
    exit 2
fi

if not fgrep -x updtools $tmpSRVtools >/dev/null 2>&1; then
    echo "$CMD: ERROR: Essential tool 'updtools' not being served -- aborting!"
    exit 2
fi

{
    #供給されたツールに対するファイル名を得る
    { TOOLS=$HOME/TOOLS/adm/upd1.1/bin/prpkg -ir < $tmpSRVtools ||
                                        echo "prpkg $?" >$errfile ;} |
    #分類順に入れる
    { sort -u                           || echo "sort $?" >$errfile ;} |
    #リストを完成するためにディレクトリを加える
    { $TADMRUG/bin/dirfillout           || echo "dirfillout $?" >$errfile ;} |
    #サーバが無視したファイルを取り除く
    { sed -f $tmpSignore                || echo "sed $?" >$errfile ;} |
    #所定のファイルに対するｐｌｉｓｔを得る
    { time $TADMRUG/bin/mkplist -rs || echo "mkplist $?" >$errfile ;} |
    #分類された順に入れる
    { sort -u                           || echo "sort $?" >$errfile ;}
    echo "$EOF"
} > $SRVplist

if test -s $errfile; then
    cmsg "Failure preparing server checksums" "$(< $errfile)"
    exit 2
fi
fi
if not plistOK $SRVplist; then
    echo "$CMD: ERROR: Corrupted plist file '$SRVplist' -- aborting!"
    exit 2
fi
rm -f $NEWFILES                         #新しいファイルの先のリストを取り除く
if not $USEPLIST; then
    > $RJE/UpdateStarted
fi
```

【００８０】

55 56

```
OVERTHRESHOLD=false
LONGTERMPROB=false
MISSINGREPORT=false
SUBSCRIBE_ERROR=false
echo "\nCurrent update threshold: $THRESHOLD"
echo "\nThe clients below are processed according to the sizes of their cascades,"
echo "largest first."
for CLIENTID in $(cfleanlsort $CLIENTLIST); do
        CLIENTID=$(take 8 $CLIENTID)
        cleanupclient $CLIENTID $YRDAY
        SUSPENDED=$LINKDIR/client/$CLIENTID/$suspended
        if test -f $SUSPENDED; then
          echo "\n---> Client $CLIENTID currently suspended"
        else
          EXPPWD=$LINKDIR/client/$CLIENTID/.pwd
          if test -s $LINKDIR/client/$CLIENTID/nidname; then
                nisid="-u$(< $LINKDIR/client/$CLIENTID/nidname)"
          else
                nisid=
          fi
          CNTplist=$RJE/$CLIENTID.$YRDAY
          CNTpkgfile=$RJE/$CLIENTID.pkg
          if not plistOK $CNTplist; then
                if test ! -f $CNTplist; then
                  MISSINGREPORT=true
                  echo "\n---> No plist file found for $CLIENTID\n"
                  integer DAYSOLD=0
                  if test ! -f $CNTpkgfile; then
                        echo "    No package file found for $CLIENTID"
                  elif not $SENDUPDATES_DORMANT && test -s "$(find $CNTpkgfile -mtime -3 -print 2>/dev/null)"; then
                        DAYSOLD=4
                        while test -s "$(find $CNTpkgfile -mtime +$DAYSOLD -print 2>/dev/null)"; do
                          ((DAYSOLD += 1))
                        done
                        ((DAYSOLD -= 1))
                        echo "    This client hasn't reported for $DAYSOLD days!"
                  fi
                  if ((DAYSOLD != 0)); then
                        ADMINDATA=$LINKDIR/client/$CLIENTID/admindata
                        if test -f $ADMINDATA; then
                          typeset NAME EMAIL
                          while read FIELD VALUE; do
                                case "$FIELD"
                                in  NAME)       NAME="$VALUE"
                                ;;  EMAIL)      EMAIL="$VALUE"
                                esac
                          done < $ADMINDATA
                          if [[ "$EMAIL" != @(none|NONE|"") ]]; then
                                {
                                  echo "Subject: expcools errors"
                                  echo
                                  if [[ "$NAME" != @(none|NONE|"") ]]; then
                                        echo "To $NAME:"
                                        echo
                                  fi
                                  echo "The RUG update code has detected an error.  Your client '$CLIENTID' has not"
                                  echo "reported to its server '$LOCALID' for the last $DAYSOLD days.  Please check"
                                  echo "your system to see what is causing this problem.  Consult the 'rugadm' HELP"
                                  echo "subsystem for advice.  The 'rugcheck' command may also be helpful"
                                  echo "in identifying the cause of this problem."
                                  echo
```

【0081】

40

57 58

```
                echo "\t\t\tSLOCALID Exptools update routine"
            } | mail $EMAIL
            echo "    Warning notice sent to $EMAIL."
          else
            echo "    No administrator email address on file for '$CLIENTID'."
            echo "    Warning notice NOT SENT to that machine's Exptools administrator."
          fi
        fi
        if ((DAYSOLD%5 == 0)); then
          {
            echo "Subject: exptools client trouble"
            echo
            echo "The RUG update code has detected an error.  The client '$CLIENTID' has not"
            echo "reported to its server '$LOCALID' for the last $DAYSOLD days.  Please contact"
            echo "the exptools administrator of that system to see what might be causing this"
            echo "problem.  You can consult the 'rugadm' HELP subsystem for advice."

            echo
            echo "\t\t\tSLOCALID Exptools update routine"
          } | mail $ADMIN_EMAIL
          echo "    Warning notice sent to $ADMIN_EMAIL."
        fi
      fi
    else
      echo "\n---> Corrupted plist file found for $CLIENTID\n"
    fi
  elif not pkgfileOK $CNTpkgfile; then
    if test ! -f $CNTpkgfile; then
      MISSINGREPORT=true
      echo "\n---> No package file found for $CLIENTID\n"
    else
      echo "\n---> Corrupted package file found for $CLIENTID\n"
    fi
  else
    echo "\nProcessing $CLIENTID `date`"
    sleep 10    # 他のプロセスがクリーンアップするのを許容する
    > $errfile
    if test ! -f $tmpSRVtools; then
      {
        mktoollist $SUBSCRLIST     || echo "mktoollist $?" > $errfile
      } > $tmpSRVtools

      if test -s $errfile; then
        cssig "Failure preparing server tool list" "$(< $errfile)"
        rm -f $tmpSRVtools
        exit 2
      fi

      if test ! -s $tmpSRVtools; then
        echo "$CMD: ERROR: No tools being served by this server -- aborting!"
        exit 2
      fi
      if not (grep -x updtools $tmpSRVtools > /dev/null 2>&1; then
        echo "$CMD: ERROR: Essential tool 'updtools' not being served -- aborting!"
        exit 2
      fi
    fi
    extractsection 1 $CNTpkgfile > $tmppkgfile
    extractsection 2 $CNTpkgfile |
    mksed > $tmpCignore
```

【００８２】

59 60

```
if test ! -s $tmppkgfile; then
   echo "$CMD: ERROR: No tools being requested by this client -- skipping!"
   SUBSCRIBE_ERROR=true
   continue
fi

if not fgrep -x updtools $tmppkgfile >/dev/null 2>&1; then
   echo "$CMD: ERROR: Essential tool 'updtools' not requested -- skipping!"
   SUBSCRIBE_ERROR=true
   continue
fi

{
   #予約されたツールを得る
   { mktoollist $tmppkgfile || echo "mktoollist $?" >$errfile ;} |
   #提出されたツールを交差する
   { comm -12 - $tmpSRVtools || echo "comm $?" >$errfile ;} |
   #ファイル名を得る
   { TOOLS=$HOME $TOOLS/sdm/upd1.1/bin/prpkg -ir ||
                                        echo "prpkg $?" >$errfile ;} |
   #分類された順に入れる
   { sort -u              || echo "sort $?" >$errfile ;} |
   #失われたｄｉｒｓをリストに加える
   { $TADMRUG/bin/dirfillout || echo "dirfillout $?" >$errfile ;} |
   #クライアントが無視したファイルを取り除く
   { sed -f $tmpCIgnore       || echo "sed $?" >$errfile ;} |
   #ｗ／ｏ　ｃｓｕｍｓｗｏを作るためにｐｌｉｓｔを入力
   { $TADMRUG/bin/mkplist -c  || echo "mkplist $?" >$errfile ;} |
   #分類された順に入れる
   { sort -u              || echo "sort $?" >$errfile ;} |
   { /usr/bin/join -j 1 -t'          ' - $SRVplist       ||
                                        echo "join $?" >$errfile ;}
} > $tmpCNTmodel

if test -s $errfile; then
   emsg "Failure analyzing client $CLIENTID checksums" "$(< $errfile)"
   continue
fi

if test ! -s $tmpCNTmodel; then
   echo "$CMD: ERROR: No files being requested by this client -- skipping!"
   SUBSCRIBE_ERROR=true
   continue
fi

CleanComm < $CNTplist |              #コメントを消す
sort -u > $tmpplist                  #ファイルが分類されたことを確認
cut -f1 $tmpplist |                  #クライアントのファイル名を得る
sed -f $tmpSignore |                 #サーバが無視したファイルを取り除く
/usr/bin/join -j 1 -t'    ' - $tmpplist > $tmpCNTplist

$TADMRUG/bin/diffplist -m $tmpCNTmodel -s $tmpCNTplist > $tmpdiff
PROBFILE=$RJE/.prob.$CLIENTID
cc=`wc -l < $tmpdiff`
if fgrep -x -e "~ .profile" $tmpdiff > /dev/null 2>&1; then
   echo "$CMD: ERROR: Update requests for this client corrupted -- skipping!"
elif [ "$cc" -gt $THRESHOLD ]; then
   echo "\t----> Too many updates for $CLIENTID: $cc -- skipping\n"
   if test -f $PROBFILE; then
         if test -z "$(find $PROBFILE -mtime -3 -print 2>/dev/null)"; then
            integer DAYSOLD=4
```

【００８３】

61 62

```
                while test -n "$(find $PROBFILE -mtime +$DAYSOLD -print 2>/dev/null)"; do
                    ((DAYSOLD += 1))
                done
                ((DAYSOLD -= 1))
                if ((DAYSOLD > 7)); then
                    LONGTERMPROB=true
                    STARS="*** "
                else
                    STARS=""
                fi
                echo "\t    ${STARS}This client has been over threshold for $DAYSOLD days!"
              fi
            else
              > $PROBFILE
              OVERTHRESHOLD=true
            fi
          elif [ "$cc" -eq 0 ]; then
            rm -f $PROBFILE
            echo "\tNo updates for $CLIENTID"
          else
            rm -f $PROBFILE
            echo "\tChanges for $CLIENTID: " $cc
            sendaddfiles $CLIENTID $tmpdiff
            sendrmndirs  $CLIENTID $tmpdiff
            sendchmods   $CLIENTID $tmpdiff
            sendrcpfiles $CLIENTID $tmpdiff
            dirlog       $CLIENTID $tmpdiff
            updlog       $CLIENTID $tmpdiff
          fi
          cp $tmpdiff $RJE/$CLIENTID.diff
        fi
      fi
    done
    SALUTATION=false
    if $OVERTHRESHOLD; then
        salutation
        cat <<-!

**********************************************************************
システムが夜毎に更新されるしきい値を超えました．$THRESHOLDより多くの
システムの変更は、exptools管理者の承認なしには更新されません．
このような承認を発行する方法は、rugadm　HELPスクリーン
「クライアントマシンへの更新ファイルの再送の承認」を参照下さい．
**********************************************************************
        !
    fi
    if $LONGTERMPROB; then
        salutation
        cat <<-!

**********************************************************************
システムが７日またはそれより多くのしきい値となりました．これらのシステムが
更新を何故得られないかを判定して下さい．
**********************************************************************
        !
    fi
    if $MISSINGREPORT; then
        salutation
        cat <<-!
```

40

【００８４】

63 64

```
**********************************************************************
今日の更新処理に含まれる夜毎の報告を時間通りにしないクライアントがあります．
この問題に対処するために「クライアント報告のチェック」の
ｒｕｇａｄｍ　ＨＥＬＰスクリーンを参照下さい．
**********************************************************************
        !
  fi
  if $SUBSCRIBE_ERROR; then
        saluration
        cat <<-!

**********************************************************************
予約リストに関して問題のあるクライアントがあります．この問題に対処するために
「ＲＵＧ　ＥＲＲＯＲメッセージの理解」のｒｕｇａｄｍ　ＨＥＬＰスクリーンを
参照下さい．
**********************************************************************
        !
  fi
  if not $USEPLIST; then
        > $RJE/UpdateEnded
  fi
elif test -s $NEWFILES; then
  echo "\nSending out newly arrived files ..."
  cp $NEWFILES $NEWFILESCOPY
  {
        cat $EXCLUDELIST
        genfilekdirs $LOCALID
  } | mksed -w > $tmpSignore
  for CLIENTID in $CLIENTLIST; do
        EXPPWD=$LINKDIR/client/$CLIENTID/.pwd
        if test -s $LINKDIR/client/$CLIENTID/uidname; then
          altuid="-u$(< $LINKDIR/client/$CLIENTID/uidname)"
        else
          altuid=
        fi
        CNTpkgfile=$RJE/$CLIENTID.pkg
        if not pkgfileOK $CNTpkgfile; then
          if test ! -f $CNTpkgfile; then
                echo "\n---> No package file found for $CLIENTID\n"
          else
                echo "\n---> Corrupted package file found for $CLIENTID\n"
          fi
        else
          echo "\nProcessing $CLIENTID `date`"
          > $errfile
          if test ! -f $tmpSRVtools; then
                {
                  mktoollist $SUBSCRLIST || echo "mktoollist $?" >$errfile
                } > $tmpSRVtools
          fi

          if test -s $errfile; then
                emsg "Failure preparing server tool list" "$(< $errfile)"
                rm -f $tmpSRVtools
                exit 2
          fi

          if test ! -s $tmpSRVtools; then
                echo "$CMD: ERROR: No tools being served by this server -- aborting!"
                exit 2
          fi
```

【００８５】

65 66

```
        extractsection 1 $CNTpkgfile > $tmppkgfile
        extractsection 2 $CNTpkgfile |
        mksed > $tmpCignore                    # クライアントが無視したファイル．ｓｅｄｓｃｒ

        if test ! -s $tmppkgfile; then
            echo "$CMD: ERROR: No tools being requested by this client -- skipping!"
            continue
        fi

        sed -f $tmpCignore $NEWFILES |   # クライアントが無視したファイルを取り除く
        TOOLS=$HOME $TOOLS/adm/upd1.1/bin/pkgpath -a |  # Get toolnames
        sort -u > $tmpCNTmodel                 # 新しいツールを分類する：ファイルリスト

        {
            # 予約されたツールを得る
            { mktoollist $tmppkgfile || echo "mktoollist $?" >$errfile ;} |
            # 提供されたツールを交差
            { comm -12 - $tmpSRVtools  || echo "comm $?" >$errfile ;} |
            # 分類された順に入れる
            { sort -u                  || echo "sort $?" >$errfile ;} |
            # 新しいツールを結合：ファイルリスト
            { /usr/bin/join -t: - $tmpCNTmodel ||
                                          echo "join $?" >$errfile ;} |
            # ファイル名を抽出
            { cut -d: -f2              || echo "cut $?" >$errfile ;} |
            # 分類された順に入れる
            { sort -u                  || echo "sort $?" >$errfile ;} |
            # 更新フォームに入れる
            { sed 's/.*/1 &/'          || echo "sed $?" >$errfile ;}
        } > $tmpdiff

        if test -s $errfile; then
            cmsg "Failure analyzing client $CLIENTID checksums" "$(< $errfile)"
            continue
        fi

        cc=`wc -l < $tmpdiff`
        if [ "$cc" -eq 0 ]; then
            echo "\tNo updates for $CLIENTID"
        else
            echo "\tChanges for $CLIENTID: " $cc
            sendrepfiles $CLIENTID $tmpdiff
            updlog       $CLIENTID $tmpdiff
        fi
        cp $tmpdiff $RJE/$CLIENTID.diff2
      fi
    done
    if cmp -s $NEWFILES $NEWFILESCOPY; then
        rm $NEWFILES $NEWFILESCOPY
    else
        comm -13 $NEWFILES $NEWFILESCOPY > $tmpfiles
        mv $tmpfiles $NEWFILES
        rm $NEWFILESCOPY
    fi
else
    echo "\nNo reason to send files, none will be sent!"
fi

echo "\nFinish `date`"

$debug cat $outfile >> $LINKDIR/server/Runlog
```

【図面の簡単な説明】

【図１】本発明の原理が好適に実施される、典型的な階層的な通信ネットワークのブロックダイヤグラムである。

【図２】本発明が実施される適切な環境を提供するとともに図１の通信ネットワークにしたがって動作する、処理システムのノードの等大の説明図である。

【図３】図１の処理システムに好適に関連付けされるとともに本発明が実施され動作される好適な環境を提供する、マイクロプロセシング回路の一例の高レベルのブロックダイヤグラムである。

【図４】図１の通信ネットワークの単一の例示のブランチの高レベルのブロックダイヤグラムである。

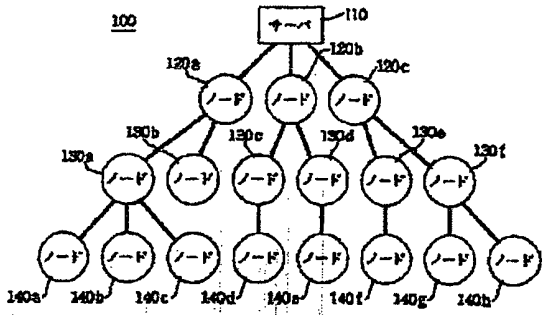
【図５】本発明の原理にしたがって図１の通信ネットワークを通って改訂を伝播するための例示の方法の動作を示したフローダイヤグラムである。

【符号の説明】

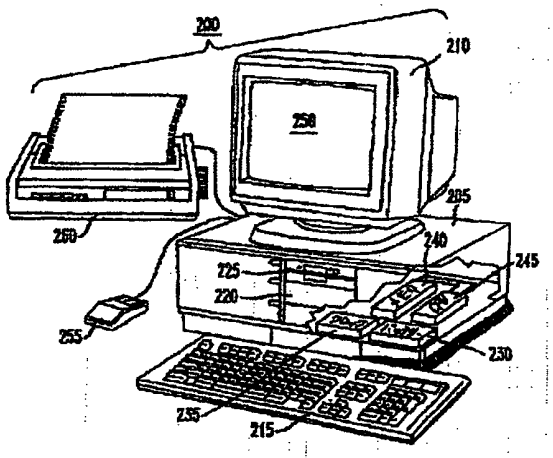
１００　通信ネットワーク

110　サーバノード
120a－120c、130a－130f、140a－140h　クライアントノード
230　バッテリー
235　クロック

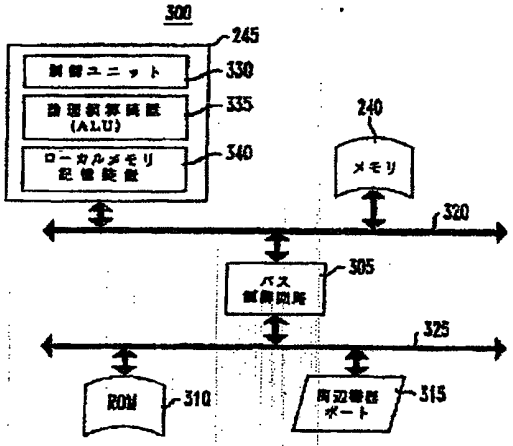
【図1】



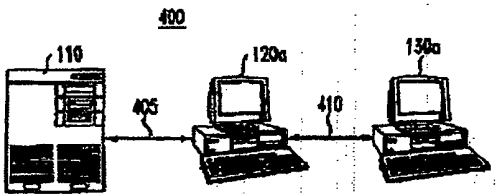
【図2】

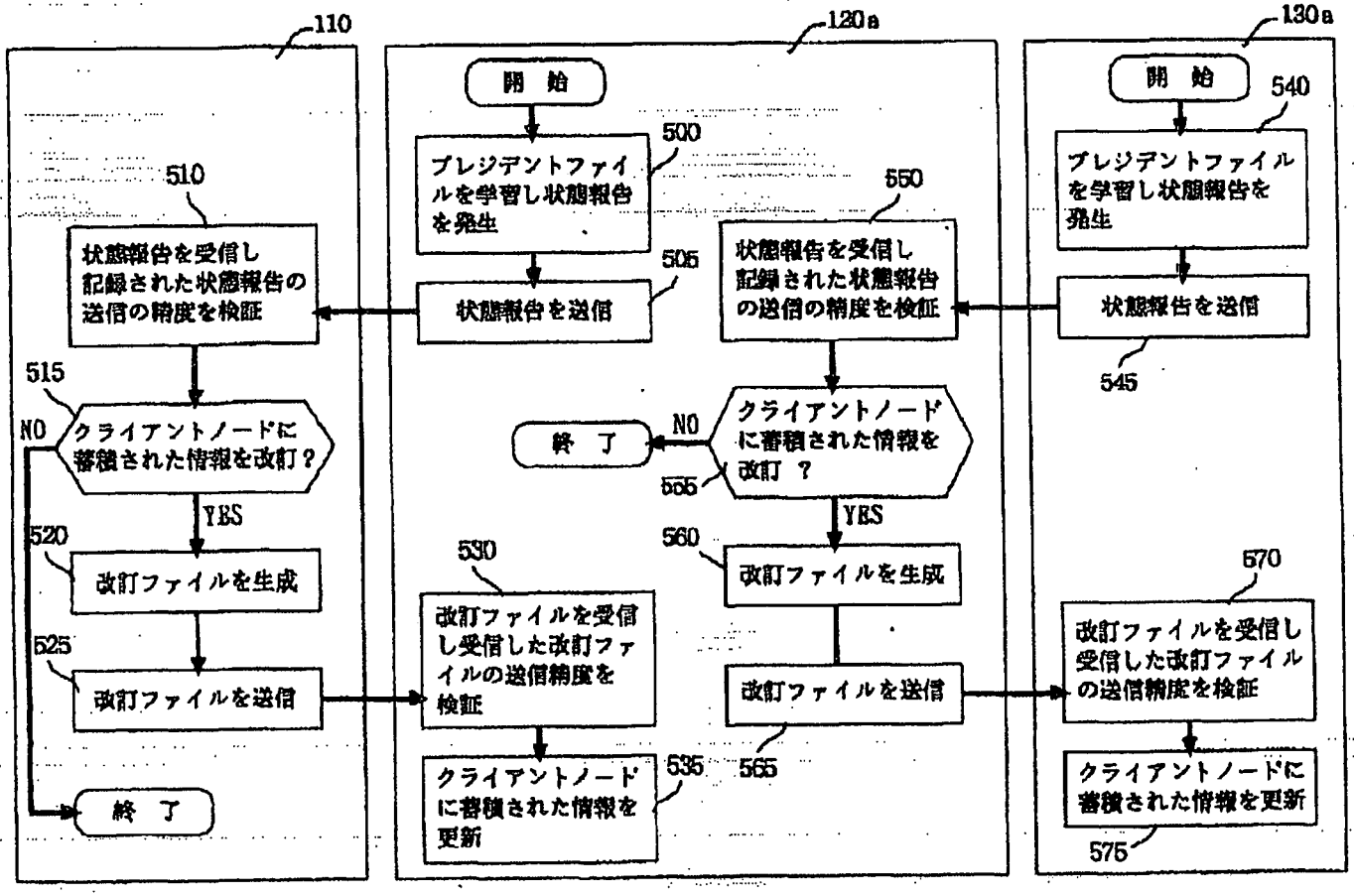


【図3】



【図4】





【図5】

特開平9－200269

This Page is Inserted by IFW Indexing and Scanning Operations and is not part of the Official Record

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ BLACK BORDERS
- ☐ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☐ FADED TEXT OR DRAWING
- ☐ BLURRED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☑ COLOR OR BLACK AND WHITE PHOTOGRAPHS
- ☐ GRAY SCALE DOCUMENTS
- ☑ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☐ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: _____

IMAGES ARE BEST AVAILABLE COPY.
As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.